С.Н.Флоренцев, В.Ю.Олунин, В.Ф.Турчин
/г.Москва/

РЕФАЛ - ИНТЕРПРЕТАТОР

Первое сообщение о языке РЕФАЛ (алгоритмический язык рекурсивных функций) появилось в [1], где язык РЕФАЛ рассматривался как несколько измененный вариант предложенного ранее метаалгоритмического языка [2,3].

В настоящем докладе описывается транслятор-интерпретатор языка РЕФАЛ для вычислительной машины БЭСМ-6. При создании транслятора пришлось ввести некоторые количественные ограничения на элементы языка, вытекающие из конечности памяти машины.

Алгоритмы, записанные на РЕФАЛе выполняются в режиме интерпретации. Однако язык имеет удобный способ введения компилированных кусков программы, что позволяет нам говорить о системе автоматического программирования с РЕФАЛа. В дальнейшем такие куски программы, написанные на языке машины-исполнителя, мы будем называть "машинными командами". Библиотека таких "машинных команд" в принципе ничем не ограничена и в дальнейшем будет расширяться по мере развития системы.

I. Краткое описание РЕФАЛа *)

Язык РЕФАЛ по существу является определенной системой описания алгоритмов преобразования символьной информации на метаалгоритмическом языке и может поэтому рассматриваться как сужение последнего.

*) Это описание отличается от данного в [1] отсутствием свободной переменной типа цепочки и незначительным изменением формы записи.

Знаки языка **РЕФАЛ** делятся на собственные и несобственные. Собственные знаки это знак ' (кавычка), круглые скобки: (,) и знаки, обозначающие управляющие символы (см.ниже). Несобственные знаки вводятся программистом по своему усмотрению. Остальные синтаксические понятия описываются следующей бэкусовской нормальной формой.

<Символ> ::= <скобка> | <управляющий символ> | <значащий символ>

<скобка> ::= (|)

<управляющий символ> ::= $ | K | S | W | E | ≥ | R | :=

<значащий символ> ::=

<несобственный знак> |

'<несобственная цепочка>'

<несобственная цепочка> ::= <несобственный знак> |

<несобственный знак> <несобственная цепочка>

<цепочка> ::= <пусто> | <цепочка> <символ>

<терм> ::= <значащий символ> | (<выражение>)

<выражение> ::= <пусто> | <выражение> <терм>

<предложение> ::= $ <примечание> <указатель направления>

K <левая часть> ≥ <правая часть>

<примечание> ::= <пусто> | <несобственная цепочка>

<указатель направления> ::= <пусто> | R

<пусто> ::=

<левая часть> ::= <цепочка>

<правая часть> ::= <цепочка>

Символ является минимальной семантической единицей, не расчленной в процессе работы машины. Описание алгоритма состоит, по существу, в определении некоторого числа рекурсивных функций,

определенных на множестве цепочек символов (цепочек) и принимающих значения из того же множества. Так как скобки входят у нас в число символов (что необходимо для описания объектов, имеющих структуру дерева), обычное функциональное обозначение $f(\xi)$, где (ξ) - некоторая цепочка, становится непригодным. Вместо $f(\xi)$ будем писать $\underline{K} f \xi .$.

Символ конкретизации $\underline{K}$ является указанием для машины, что следующую за ним цепочку надо рассматривать как функциональное обозначение и "конкретизировать" ее, т.е. вычислить значение функции. Конец этой цепочки (которую **назовем областью** действия символа конкретизации) отмечается подчеркнутой точкой. При обозначении функции от функции символ и точка выступают в роли левой и правой скобок.

Если цепочка содержит более одного символа $\underline{K}$, то один из **них назовем "ведущим".** Он определяется как самый левый символ $\underline{K}$ из всех символов $\underline{K}$, в области действия которых нет других $\underline{K}$. Например, обычной записи

(1) $f(a, h(g(x), p(y)))$

будет соответствовать запись $\underline{K} f a, \underline{K} h \underline{K} g x ., \underline{K} P y . . .$ и ведущим символом $\underline{K}$ будет символ $\underline{K}$, стоящий перед буквой g.

Цепочка, обрабатываемая машиной, помещается в поле зрения машины. Кроме поля зрения машина имеет поле памяти, содержащее список предложений. Предложения описывают правила конкретизации, которые в языке РЕФАЛ сводятся к замене одной цепочки на другую. Левая часть предложения не может содержать символов $\underline{K}$ (и точки, разумеется), но может содержать свободные переменные, для изображения которых служат указатели переменных: ука-

затель символа $\underline{S}$, указатель терма (слова) $\underline{W}$, указатель выражения $\underline{E}$. Соответственно этому, свободные переменные могут быть трёх типов и изображаются соответствующим указателем и непосредственно следующим за ним **значащим символом** (играющим роль индекса в обычных обозначениях), например: $\underline{S}1$, $\underline{E}$ <конец> и т.п. Предложение является применимым для конкретизации данной цепочки, если эта цепочка может быть отождествлена как левая часть предложения, т.е. свободным переменным, входящим (быть может) в левую часть, могут быть приданы такие значения (с соблюдением синтаксического вида), что левая часть совпадет с цепочкой. Применение предложения заключается в замене конкретизируемой цепочки вместе с символом $\underline{K}$ и (подчеркнутой) точкой на правую часть предложения, в которой переменные заменены **их значениями.** Если существует несколько вариантов придания значений свободным переменным, приводящих к отождествлению, то выбор одного из них зависит от указателя направления, входящего в предложение, и осуществляется **следующим образом. Если** указатель направления пустой, **то из всех вариантов выбирается тот, при котором самая левая свободная переменная выражения на основном уровне скобочной структуры принимает значение, содержащее минимальное число символов. Если это не устраняет неоднозначности, то такой же отбор проводится по остальным переменным основного уровня в порядке слева направо. Если же и это не устраняет неоднозначности, то сравниваются свободные переменные выражений на следующем уровне скобочной структуры и так далее. Если указатель направления есть** $\underline{R}$, то аналогичная процедура проводится справа налево. Например, предложение

(2) $\S\underline{K}\, f\, \underline{E}1, \underline{E}2 \geq \underline{E}1$

определяет функцию f, **которая выделяет часть цепоч-**

ки до первой запятой, а предложение

(3) $\S \underline{R}\, \underline{K}\, g\, \underline{E}^1, \underline{E}^2 \geq \underline{E}^1$ —

функцию g , выделяющую отрезок до последней запятой (первой справа). В обоих случаях просмотр совершается без вхождения в скобки.

Машина работает по шагам. Каждый шаг начинается с отыскания ведущего символа конкретизации. Если его область действия оказывается "машинной командой", то она выполняется. В противном случае машина просматривает список предложений сверху вниз и применяет первое из предложений, которое оказалось применимым. Затем она выполняет следующий шаг, и т.д. Когда в поле зрения не оказывается ни одного символа $\underline{K}$, машина останавливается.

2. Основные принципы интерпретатора

При создании транслятора основное внимание уделялось повышению эффективности режима интерпретации. Легко видеть, что буквальное следование описанию семантики языка приводит к многократному просмотру и переписи участков обрабатываемого текста.

Описанные ниже принципы построения интерпретатора позволяют ликвидировать практически все лишние, т.е. не вытекающие из самой сущности алгоритма, просмотры и переписи.

2.I. Система детерминативов

В соответствии с описанием языка для отыскания подходящего предложения машина просматривает все предложения сверху вниз, проводя каждый раз синтаксическое отождествление. Нетрудно заметить, что при наличии большого количества предложений машина будет делать много лишних просмотров, прежде чем найдет необходимое предложение.

С другой стороны, программу на РЕФАЛе можно представлять как набор рекурсивных функций, каждая из которых описывается одним или некоторым количеством предложений. У предложений,

описывающих одну и ту же функцию, все левые части начинаются с одного и того же символа - указателя функции (в дальнейшем будем называть его детерминативом).

Все предложения с одним и тем же детерминативом в трансляторе связаны в цепочку просматриваемых предложений. Адрес самого верхнего из них указывается в таблице детерминативов. Транслятор, выбрав символ, стоящий в поле зрения после символа $\underline{K}$, просматривает только те предложения, у которых детерминатив равен данному символу. Более подробно о построении цепочки просматриваемых предложений будет сказано ниже.

2.2. Расположение информации в памяти машины

Для нахождения переменных типа $\underline{W}$ или $\underline{E}$ в соответствии с буквальным описанием синтаксического отождествления необходимо посимвольно набирать содержимое указанных переменных, двигаясь в поле зрения слева направо (или справа налево). Для того, чтобы иметь возможность определять содержимое переменной типа $\underline{W}$ и в некоторых случаях (см.ниже) переменной типа $\underline{E}$ за минимально возможное время, поле зрения в трансляторе организовано следующим образом.

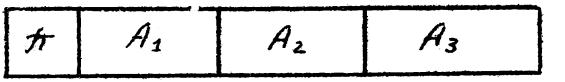

Рис. I.

Ячейка машины БЭСМ-6 делится на четыре части (см.рис.I), где A_i составляет I5 разрядов (в соответствии с адресностью машины БЭСМ-6). Каждая ячейка помещает один символ. Признак

$\mathcal{f}$ определяет, является ли этот символ значащим символом, скобкой или символом конкретизации $\underline{K}$. От содержания $\mathcal{f}$ зависит содержание A_1, A_2, A_3.

I. Если $\mathcal{f}$ соответствует значащему символу, то в A_2 хранится код символа (9 разрядов), в A_1 - адрес предыдущего, а в A_3 - последующего символа.

2. Если $\mathcal{f}$ соответствует скобке, то в A_2 хранится адрес парной скобки, в A_1 и A_3 – адрес предыдущего и последующего **символов.**

3. Если $\mathcal{f}$ соответствует символу конкретизации, то в A_1 и A_3 хранится адрес предыдущего и последующего символа, а в A_2–адрес того символа $\underline{K}$,который станет ведущим, когда будет полностью выполнена данная конкретизация.

Каждый символ в трансляторе кодируется 9-ю разрядами. И предложения общей памяти записываются в сжатом виде по 5 символов в ячейке.

Кроме этого в трансляторе организуется массив свободного поля памяти (СПП), который устроен так же, как и поле зрения.

Такое расположение информации в памяти машины позволяет целиком избежать посимвольного просмотра при наборе содержимого $\underline{W}$. Переменная $\underline{W}$ *) набирается за один шаг,используя адрес связи A_2 парной скобки.

---

*) Для запоминания содержимого переменных $\underline{S}$, $\underline{W}$ и $\underline{E}$ транслятор отводит соответствующую таблицу, Эта таблица называется таблицей регистров. Поэтому в дальнейшем часто будем говорить о переменных, как о соответствующих регистрах.

Что же касается регистров типа $\underline{E}$, то в трансляторе различаются два типа регистров: так называемый открытый и закрытый регистр $\underline{E}$.

Тип регистра $\underline{E}$ в данном предложении тесно связан со скобочной структурой его левой части.

Если на нулевом уровне скобочной структуры нет ни одного регистра $\underline{E}$ либо он всего один, то такая скобочная структура называется закрытой. И регистр $\underline{E}$ в этом случае мы также называем закрытым. Если регистров $\underline{E}$ несколько, то скобочная структура называется открытой. И все регистры $\underline{E}$, кроме самого правого (или левого) называются открытыми. Самый правый регистр $\underline{E}$ (или левый при отождествлении справа налево) является закрытым.

Пример (4 показывает одно из предложений, описывающих на языке РЕАЛ процедуру α.

(4) $\S \underline{K}\alpha\, \underline{E}a\,(\underline{E}b + \underline{E}c)\,\underline{E}d \geq \underline{E}a\,(\underline{E}b - \underline{E}c)\,\underline{K}\alpha\, \underline{E}d.$

В соответствии с описанием языка данное предложение будет работать до тех пор, пока в поле зрения будет находиться хотя бы одна скобочная структура, внутри которой есть хотя бы один знак + . В приведенном примере открытыми регистрами являются $\underline{E}a$, $\underline{E}b$, а закрытыми $\underline{E}c$ и $\underline{E}d$. Скобочная структура имеет два уровня *).

Самая внешняя скобочная структура является открытой. Это определяется только тем, что на её нулевом уровне находятся два регистра $\underline{E}a$ и $\underline{E}d$. Наличие регистров $\underline{E}b$ и $\underline{E}c$ на следующем уровне никак не влияет на определение типа скобочной

---

*) Символы $\underline{K}$ и $\geq$ понимаются транслятором как левая и правая скобки.

структуры. Это определение рекурсивно переносится на следующий уровень скобочной структуры.

Тот факт, что транслятор различает два типа регистров $\underline{\underline{E}}$, позволяет ему избежать лишних хождений в поле зрения для набора содержимого закрытого регистра $\underline{\underline{E}}$. В этом случае транслятор обращается к скобке, закрывающей соответствующую скобочную структуру, меняет направление отождествления на противоположное и после того, как вновь дойдет до данного регистра $\underline{\underline{E}}$, оставшуюся часть поля зрения относит к содержимому указанного регистра.

Для набора открытого регистра $\underline{\underline{E}}$ необходимости просмотра поля зрения не избежать. Однако встречающиеся при этом скобочные структуры проходятся за один шаг, используя адреса связи у парных скобок.

И здесь следует заметить, что необходимость просмотра поля зрения не является какой-то потерей эффективности в трансляторе. Дело в том, что в РЕФАЛе наличие открытой скобочной структуры как правило всегда означает некий просмотр поля зрения. Содержимым открытого регистра $\underline{\underline{E}}$ является как раз просмотренная в результате отождествления часть поля зрения. Закрытый же регистр $\underline{\underline{E}}$, представляющий оставшуюся непросмотренную информацию, транслятором и не просматривается.

2.3. Отождествление скобочных структур

Многоуровневые скобочные структуры отождествляются в трансляторе следующим образом.

Отождествление любой скобочной структуры ведется только на нулевом уровне. Все внутренние скобочные структуры при этом пропускаются. В случае удачного отождествления вступает в силу

принцип последовательного приоритета самой левой (или правой при отождествлении справа налево) закрытой скобочной структуры. Приоритет этот не распространяется на закрытые скобочные структуры, находящиеся внутри открытых скобочных структур.

После того, как в левой части предложения останутся только одни открытые скобочные структуры, приоритет отдается самой левой (соответственно правой) открытой скобочной структуре.

В случае неудачного отождествления закрытой скобочной структуры отождествление **всего** предложения считается неуспешным, если только данная структура не находится внутри какой-либо открытой скобочной структуры. В последнем случае управление передается на удлинение соответствующего открытого регистра.

При неудачном отождествлении открытой скобочной структуры управление передается на удлинение самого правого из числа отождествленных (соответственно **левого**) регистров ,если только данная структура не находится внутри какой-либо открытой скобочной структуры. В последнем случае управление передается на удлинение самого правого (соответственно **левого**) открытого регистра $\mathcal{E}$ во внешней по отношению к данной скобочной структуре.

Отождествление считается успешным после того, как отождествятся все скобочные структуры.

Приведем пример сложной скобочной структуры в левой части предложения. Закрытые скобочные структуры заключим в квадратные скобки, а открытые-в круглые [36].

---

[36] В РЕФАЛе для обозначения скобок допустимы только круглые скобки. Квадратные введены только для наглядности.

$(5)\ \S\ \underline{K}\alpha[_1(_2[_3\ ]_3)_2]_1\ (_4\ )_4\ [_5[_6\ ]_6]_5 \geq$

Порядок отождествления скобочных структур в примере (5) будет следующим: сначала отождествление ведётся на нулевом уровне самой внешней скобочной структуры и далее $[_1\ ]_1$, $[_5\ ]_5$, $[_6\ ]_6$, $(_2\ )_2$, $[_3\ ]_3$, $(_4\ )_4$.

2.4. Замена, „чистильщик"

При замене левой части предложения на правую все управляющие и значащие символы записываются в ячейках СПП. Содержимое же регистров не переписывается, а используется информация таблицы регистров.

В трансляторе число различных регистров в левой части ограничено I6-ю. В соответствии с этим таблица регистров содержит I6 ячеек. Для каждого регистра в ячейках этой таблицы указывается адрес начала и конца содержимого свободной переменной, представленной данным регистром.

Когда в правой части предложения встречается какой-либо регистр, то происходит перекоммутация начала и конца содержимого регистра с новыми символами, стоящими перед и после данного регистра, т.е. изменяется адрес A_1 начала содержимого регистра и адрес A_3 конца регистра.

После замены левой части предложения на правую включается так называемый "чистильщик". Он формирует массив ячеек из области действия ведущего символа конкретизации, оказавшихся ненужными в процессе замены, и подшивает этот массив к началу свободного поля памяти. Это достигается за счёт перекоммутации адресов связи у соответствующих символов. При этом не происходит никаких лишних просмотров. Так, например, если в правой части предложения отсутствует какой-либо регистр, который есть

в левой части, то "чистильщик" содержимое этого регистра не просматривает,а изменяет только адреса связи начала и конца регистра.

3. Основные блоки транслятора

Транслятор с РЕФАЛа включает в себя следующие основные блоки:

I. Блок синтаксического контроля (БСК) вводимой информации.

2. Препроцессор.

3. Процессор.

4. Блок обмена с внешними ЗУ (БОВЗУ).

БСК никакой принципиальной нагрузки не несет. Лимитируемые рамками доклада,авторы вынуждены не рассматривать блок обмена с внешними ЗУ.

3.I. Препроцессор

Препроцессор состоит из блока кодировки информации во внутренний код транслятора (БКИ), блока записи предложений (БЗП) и блока записи поля зрения (БЗПЗ).

Каждое предложение пробивается на УПП на отдельной перфокарте (либо на нескольких подряд идущих перфокартах). В конце колоды подкладывается перфокарта, на которой пробит знак конца массива предложений (§§). Отдельно пробиваются выражения массива поля зрения. В конце этого массива ставится перфокарта с символом конца вводимой информации ('ФИНИШ' 0). Первым в машину вводится массив предложений, а затем поле зрения.

БКИ кодирует каждый символ языка 9-ти разрядным кодом,в частности, сложный символ, заключенный в кавычки, также кодируется 9-тью разрядами. Содержимое такого символа, представляющего собой цепочку символов, запоминается в соответствующем массиве памяти. При печати эта информация используется для восстановления такой цепочки.

Знак §, коментарий и символ $\underline{K}$ левой части предложения в поле памяти не записываются. Левая часть каждого предложения начинается с детерминатива. Здесь мы уточним понятие детерминатива.

Детерминативом называется последовательность символов, расположенных в левой части предложения после символа $\underline{K}$, до ближайшей переменной типа $\underline{S}$, $\underline{W}$ или $\underline{E}$. В настоящем трансляторе число символов, входящих в детерминатив, ограничено тремя. В частности, детерминатив может быть и пустым. Предложения с пустым детерминативом относятся препроцессором к одному типу. Такие предложения объединяются в цепочку просматриваемых предложений. Достигается это тем, что при записи каждому предложению предшествует ячейка памяти (информационная ячейка), в которой указывается адрес следующего предложения данного типа. Адрес самого верхнего предложения указывается в рабочей ячейке.

Предложения с непустым детерминативом объединяются в цепочку просматриваемых предложений следующим образом.

Пусть имеется последовательность вводимых предложений: A_1, $A_2, \dots, A_n$. Назовём предложение A_i "более общим", чем A_j , если выполняется следующее равенство:

$$Det(A_j) = Det(A_i)C \ ,$$

где $Det(A)$ обозначает детерминатив предложения A , а C является цепочкой символов, которая не может быть пустой. Например, пусть предложения A_1, A_2, A_3, A_4 имеют детерминативы (соответственно) ab , ab , abc , a , ac . Предложение A_1 является "более общим" по отношению к предложению A_3 , но не по отношению к A_2 , A_4 и A_5 .

Назовем последовательность предложений монотонной, если никакое следующее не является "более общим", чем предыдущее. Такой последовательности препроцессор сопоставляет иерархию, состоящую из узлов. В узлах содержится следующая информации (рис.2):

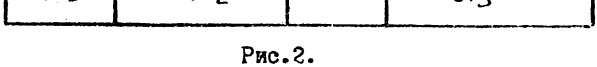

Рис.2.

u_1 – код соответствующего символа детерминатива,

u_2 – адрес узла, соответствующего предложению, детерминатив которого на один символ больше,

u_3 – адрес узла, соответствующего предложению, детерминатив которого отличается от данного последним символом.

Таким образом, адреса в узлах указывают путь от "более общего" предложения к "менее общему".

При нарушении монотонности образуется вершина новой иерархии. В таблице детерминативов указывается адрес последней из них. В информационной ячейке предложения, соответствующего такой вершине, указывается адрес вершины той иерархии, нарушением монотонности которой явилось данное предложение. В информационных ячейках остальных предложений указывается адрес начала левой части ближайшего "более общего" предложения внутри соответствующей иерархии.

Препроцессор устанавливает следующий просмотр предложений, детерминатив которых начинается с одного и того же символа. По первому символу, стоящему в поле зрения вслед за символом , в таблице детерминативов находится адрес вершины самой последней иерархии. В соответствии с последующими двумя символами поля зрения отыскивается соответствующий узел такой иерархии. Если такого узла в данной иерархии нет, то управление передается на анализ предыдущей иерархии. После того, как соответствующий узел в какой-либо иерархии найден, дальнейшее движение по иерархии в случае неудачного отождествления производится в соответствии с адресами, расположенными в информационных ячейках предложений.

При записи левой части предложения происходит анализ скобочных структур и разделение на типы регистров Σ . Сначала производится запись символов на нулевом уровне скобочной струк-

туры. Все внутренние скобочные структуры при этом пропускаются. В качестве их представителей на нулевом уровне остаются символы левых скобок. После таких скобок записываются номера, указывающие порядок отождествления соответствующих скобочных структур. Внутренние скобочные структуры записываются после внешней. При этом порядок записи таких структур определяется принципами, описанными в пп. 2,3 настоящего доклада. В результате такой записи процессор в дальнейшем отождествляет скобочные структуры последовательно слева направо, и при этом правила отождествления скобочных структур будут соблюдены.

При наличии в комментарии знака $\underline{R}$ левая часть предложения записывается в поле памяти в обратном порядке. Однако определение детерминатива при этом не меняется.

Правая часть предложения записывается после записи левой части. В информационной ячейке указывается адрес начала левой части.

Концом обрабатываемого массива информации для БЗП служит знак §§ . После этого управление передается в БЗПЗ.

БЗПЗ кодирует информацию в поле зрения транслятора так, как это описано в п.2.2 настоящего доклада. Область действия символа $\underline{K}$ заключается в скобки. Встретив символ 'ФИНИШ', БЗПЗ заканчивает работу и передает управление процессору. При этом в соответствующих рабочих ячейках указываются адреса ведущего символа конкретизации в начале СПП.

4. Процессор

Процессор состоит из 3-х основных блоков: блок поиска предложения общей памяти (БППОП), блок отождествления (БО) и блок переписи правой части (БППЧ). При входе в процессор управление передается в БППОП. Если I-й символ поля действия ведущего символа конкретизации или конкретного ряда - идентификатор машинной команды, то управление передаётся на выполнение этой команды. В противном случае, по первым трём символам конкретного ряда производится поиск первого из цепочки просматриваемых предложений. Как только найдено такое предложение, включается БО, если же поиск оказался неудачным, то управление передается на выход "Задача не решается". В БО в соответствии с алгоритмом, описанным выше, производится попытка синтаксического отождествления конкретного ряда как левой части предложения общей памяти. В случае неудачной попытки управление передается обратно в БППОП, где производится поиск следующего в цепочке просматриваемых предложений. В случае удачного отождествления управление передается в БППЧ. В этом блоке производится замена старого конкретного ряда новым, в соответствии с правой частью отождествлённого предложения общей памяти. После переписи ненужные части старого краткого ряда "подниваются" к свободной памяти. Затем, если в поле зрения ещё остались символы $\underline{K}$, то управление передается в БППОП, если же их нет, то на выход "задача окончена". Теперь рассмотрим подробнее работу БО и БППЧ.

Блок отождествления

В памяти машины левая часть предложений хранится в сжатом виде В целях повышения эффективности работы БО производится развертка левой части, при этом каждому символу отводится одна ячейка. Во время развертки в ячейки, соответствующие регистрам $\underline{E}$ и правым

заключительным скобкам (ПЗС) заносятся некоторые вспомогательные адреса, используемые в других блоках БО. Развертка производится до I-й ПЗС, так как в случае неудачного отождествления этой части развертка остальной была бы бесполезна (что следует из самого алгоритма отождествления), в случае же удачного отождествления развертка продолжается до очередной ПЗС. Развернутую часть будем называть абстрактным рядом. При отождествлении значащих символов и скобок абстрактного ряда они сравниваются с соответствующими символами конкретного ряда. При несовпадении управление передается в блок удлинения (БУ), при совпадении выбирается следующий символ абстрактного ряда. Для отождествления регистров заводятся три таблицы: T_1, T_2 и T_3. В табл. T_1 заносятся адреса начала и конца содержимого первого регистра с данным идентификатором. В таблице T_2 каждая из 16 ячеек является счётчиком числа уже отождествленных регистров с данным идентификатором. В табл. T_3 заносятся по мере отождествления идентификаторы регистров, адреса их содержимого, а для "удлиняемых" регистров (о них сказано ниже) их адрес в абстрактном ряду. При отождествлении регистров сначала по табл. T_2 устанавливается, встречался ли до этого регистр с таким же идентификатором. Если да, то по табл. T_1 находятся адреса начала и конца содержимого первого из таких регистров. Затем подлежащие отождествлению символы конкретного ряда последовательно сравниваются с содержимым I-го регистра. При совпадении всех символов выбирается следующий символ абстрактного ряда, при несовпадении управление передается в БУ. Пусть теперь регистр с данным идентификатором встречается впервые. Для регистров $\underline{S}$ проверяется, не является ли соответствующий символ конкретного ряда скобкой, а для регистров $\underline{W}$ – не является ли он закрывающей скобкой. Если является, то управление передается в БУ, если нет —

выбирается следующий символ абстрактного ряда. Отождествление закрытых регистров $\underline{\underline{E}}$ описано выше. Как и в остальных случаях при неудачном отождествлении управление передастся в **БУ**.

Впервые встретившиеся открытые регистры $\underline{\underline{E}}$ будем называть „удлиняемыми". При отождествлении таких регистров им присваивается некоторое "пробное" число символов (какое именно, сказано ниже) и производится попытка отождествить остальные символы. Пусть эта попытка оказалась неудачной и управление передалось в **БУ**. В этом блоке по **табл.** T_3 от конца её к началу производится поиск "удлиняемого" регистра (точнее ячейки, соответствующей такому регистру). Для остальных регистров, которые встречаются при этом поиске, соответствующий счётчик в T_2 уменьшается на 1. Если "удлиняемых" регистров нет, то, следовательно, отождествление невозможно, и управление передается в БППОП. Если же они есть, то производится удлинение первого найденного "удлиняемого" регистра, если это возможно (ограничением здесь является то, что регистры $\underline{\underline{E}}$ должны содержать только правильные скобочные структуры). Если же это невозможно, то удлиняется предшествующий данному "удлиняемый" регистр и т.д. Теперь рассмотрим само отождествление "удлиняемых" регистров. Опорным символом регистра $\underline{\underline{E}}$ назовем I-й справа (при движении слева направо) от него значащий символ или скобку, такой, что между ним и данным регистром $\underline{\underline{E}}$ нет других регистров $\underline{\underline{E}}$. При отождествлении "удлиняемых" регистров без опорного символа, вначале ему присваивается пустое число символов. При удлинении такому регистру присваивается соответствующий ему символ конкретного ряда, затем следующий и т.д. пока это возможно. (Естественно, если регистру присваивается левая скобка, то ему присваивается и правая, и все символы, стоящие внутри этих скобок). Если есть регистры с опорным символом, то вначале производится поиск по конкретному ряду,

на одном уровне скобочкой структуры символа, совпадающего с опорным. Затем в абстрактном ряду по адресу связи производится переход к опорному символу и отождествление ведется в обратном направлении, как и в случае закрытого регистра $\underline{\underline{E}}$. При удлинении такого регистра производится поиск следующего символа, совпадающего с опорным, до тех пор, пока такой поиск возможен. Итак, мы рассмотрели все возможные случаи отождествления. После того, как отождествление будет закончено, управление передается в БППЧ.

Блок переписи правой части

При переписи правой части предложения для каждого символа правой части заводится один или несколько символов в свободной памяти машины. Перепись значащих символов, скобок, символов $\underline{K}$ и точек производится так же,как и в препроцессоре перепись поля зрения. При переписи регистров необходимо для каждого из них поставить в свободной памяти его содержимое. Для этого по табл. T_3 производится поиск ячейки, соответствовшей регистру с таким же идентификатором. По этой ячейке определяется адрес начала и конца содержимого регистра, и затем ячейка стирается. После этого производится переадресация первого и последнего символа содержимого, они связываются с остальными новыми символами. Затем связываются символы старого конкретного ряда предшествовавшие и следующие после содержимого, в результате чего всё содержимое оказывается как бы "выкинутым" из старого конкретного ряда и"пришито" к новому. Возможно, что в табл. T_3 больше не окажется регистров с данным идентификатором, в то время как в правой части они ещё будут. В этом случае приходится переписывать всё содержимое на свободные ячейки, последовательно связывая их. Остальные функции БППЧ рассмотрены выше.

Машинные команды

Машинная команда представляет независимую подпрограмму, например, печать поля действия ведущего символа конкретизации, занесение предложения в общую память, сложение десятичных чисел, находящихся в поле действия ведущего символа конкретизации и т.п. Обычно машинные команды вводятся, когда данную операцию невозможно выполнить на уровне языка, но для повышения эффективности, некоторые операции, например, сложение десятичных чисел, выполнение которых на уровне языка возможно, но не эффективно, также вводятся, как машинные команды. В трансляторе предусмотрена возможность использования математиком собственных машинных команд. Любая машинная команда должна быть выполнена как перемещаемая программа и записана на ленту. Для того, чтобы некоторая машинная команда использовалась при решении данной задачи, необходимо вместе с задачей ввести в машину информационную перфокарту, на которой указан детерминатив машинной команды и ее адрес на ленте. Машинная команда переписывается с ленты на свободную память машины. Для обращения к машинной команде достаточно, чтобы I-й символ поля действия ведущего символа конкретизации являлся детерминативом данной машинной команды.

Весь процессор помещается в одном листе МОЗУ (I024 ячеек) машины БЭСМ-6.

Литература

I. В.Ф.Турчин, С.Н.Флоренцев, В.А.Фисун, Язык РЕФАЛ и его использование в задачах автоматизации программирования, доклад на межвузовской научной конференции "Алгоритмизация и программирование экономических расчетов", М., Май, I967.

2. В.Ф.Турчин, в об. "Цифровая вычислительная техника и программирование", изд-во "Советское радио", М.,I966.

3. В.Ф.Турчин, журн. "Кибернетика" /в печати/.

Межведомственная комиссия по математическому обеспечению

Государственного комитета Совета Министров СССР по науке и технике

ИНСТИТУТ КИБЕРНЕТИКИ АН УССР

Научный совет по комплексной проблеме "Кибернетика" АН СССР

Научный совет по кибернетике АН УССР

Киевский дом научно-технической пропаганды

ПЕРВАЯ ВСЕСОЮЗНАЯ КОНФЕРЕНЦИЯ ПО ПРОГРАММИРОВАНИЮ

В. Процессоры с известных языков

К И Е В -I 9 6 8

БФ 01988.Подписано к печати 8.X 1968 г.Изд.№1-53.
Зак.653.Объем 6,4 п.л.Тираж 1050 экз.Цена 45 коп.

Лаборатория офсетной печати ИК АН УССР.
Киев-28,Б.Китаевская,109.