УДК 681.3.06:51

ПЕРВИЧНЫЕ СРЕДСТВА ПОДГОТОВКИ СИНТАКСИЧЕСКИ ПРАВИЛЬНЫХ ПРОГРАММ И ДАННЫХ ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

И.В. Вельбицкий

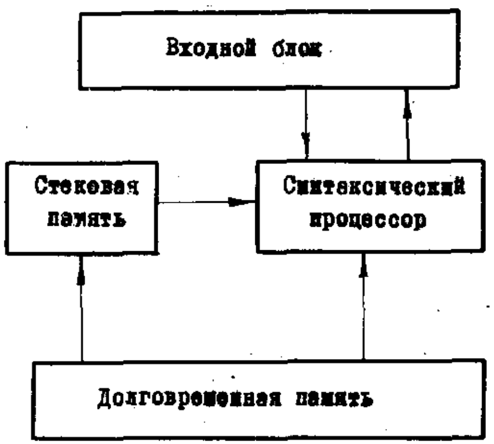
Особенностью современного этапа развития вычислительной техники является обострение противоречий, обусловленных большим разрывом в мощностях средств обработки информации и средств подготовки исходных данных. Это, а также отсутствие эффективных средств связи человека с машиной приводят к тому, что процесс отладки программ превращается в наиболее трудоемкий и длительный процесс и большие мощности вычислительных машин тратятся на его автоматизацию. Поэтому за последнее время появилось большое число терминалов, отладочных пультов, отображающих экранов и других устройств, призванных упростить связь человека с машиной.

В условиях увеличения потребителей (пользователей) вычислительных машин и систем большое значение приобретают средства входного оперативного контроля, позволяющие в процессе подготовки запросов к машине осуществлять формальную проверку их правильности. Такую проверку можно эффективно осуществить, если грамматика языка, на котором записываются исходные данные, программы и запросы к вычислительной системе заложены в постоян -

ную память некоторого специализированного устройства,непосредственно связанного со средствами, генерирующими указанную выше информацию. Такое устройство облегчает обучение человека связи с машиной или системой машин, а также значительно сокращается общее время реализации задач в системе человек-машина.

Настоящее сообщение посвящено краткому описанию устройства, предназначенного для подготовки синтаксически правильных програм и данных, а также описанию метаязыка, ориентированного на синтаксический контроль и применение в этом устройстве.

I°. Устройство состоит (см.блок-схему) из постоянной памяти (для хранения грамматик языков в описанном ниже метаязыке),входного блока (для формирования текущего символа подготавливаемой информации), синтаксического процессора(для сравнения текущего символа подготавливаемого предложения с правилами грамматики) и стековой памяти (для организации конструкций типа скобочных). В качестве устройства подготовки данных могут использоваться флексорайтеры, печатающие машинки типа "Консул", специальная клавиатура с комплектом считывающего, перфорирующего оборудования и т.д. Устройство подготовки подключается к синтаксическому процессору таким образом, что в процессе работы блокируется печать или перфорация символа, если он не соответствует правилам грамматики языка, записанным в постоянной памяти. Такое устройство позволяет осуществить подготовку синтаксически правильных програм и данных, записанных на любом



языке, грамматика которого предварительно записана в постоянную память устройства. Переориентация устройства на новый язык сводится к замене одного блока постоянной памяти другим, в котором записана грамматика нового языка.

2°. Грамматика языка в постоянной памяти устройства записывается в метаязыке, ориентированном на синтаксический контроль. Отличительной особенностью описаний грамматик в этом метаязыке является отсутствие в них в явной форме нетерминальных символов. Функции последних принимают на себя имена некоторых подмножеств правил грамматики. Благодаря этому процесс синтаксического контроля упрощается и сводится, по существу, к бесступиковому просмотру нескольких (для реальных языков, в среднем двух, трех) правил грамматики.

Пусть T -терминальный словарь, элементы которого в дальнейшем будут обозначаться малыми латинскими буквами: $\mathcal{A}$ -некоторый вспомогательный словарь, $\phi \in \mathcal{A}$; $\mathcal{R} = \{r_i\}_{i \in \Gamma}$, $\Gamma = \{1, \ldots, m\}$- множество имен некоторых конечных подмножеств (возможно,пустых) правил вида:

$$a \xrightarrow{W_S} r \,, \tag{I}$$

$$a \xrightarrow{W_S} .\, r \,, \tag{2}$$

где $a \in T$; $r \in \mathcal{R}$; имя пустого подмножества правил - ϕ ; $\xrightarrow{W_S}$ или $\xrightarrow{W_S}.$, $W_S = W_S(\nu_1, \nu_2, \ldots, \nu_n)$, $\nu_i \in \mathcal{A}$, $1 \leqslant i \leqslant n$,- многоместное отношение, означающее, что терминальный элемент слева от него некоторым образом конкатенирует с терминальным элементом левой части любого правила из подмножества, имя которого указано справа от данного W_S отношения; конкатенация терминальных элементов определяется типом S (S = 0,I,2) W_S - отношения.

Число n аргументов W_S -отношения соответствует числу стеков, используемых при интерпретации правил грамматики. При этом предполагается, что на вершине пустых стеков помещены символы ϕ .

Правило вида (I) интерпретируется следующим образом:

а) при S = 0 (W_S - отношение нулевого типа) терминальный символ a может конкатенировать с терминальными символами, встречающимися в левых частях правил из подмножества, имеюще -

го имя r ; тем самым данное правило определяет свои возможные правила - преемники из множества R , если в качестве r стоит символ $\varnothing$, символ a является последним символом предложения языка;

б) при $S = 1$ (W_S -отношение первого типа) указанная конкатенация и преемственность правил имеет место при одновременной записи тех ν_i в соответствующие i -е стеки, для которых ν_i не является символом $\varnothing$; очевидно, $W_0 \equiv W_1$,если все ν_i из W_1 являются символами $\varnothing$;,

в) при $S = 2$ (W_S -отношение второго типа) указанная конкатенация и преемственность правил, соответственно, имеет место лишь при условии совпадения всех отличных от $\varnothing$ символов ν_i с символами в вершинах соответствующих стеков, из которых при этом происходит выборка. При отсутствии такого совпадения ни конкатенация, ни преемственность правил не имеют места.

Все сказанное о правилах вида (I) имеет место и для правил вида (2) с тем лишь отличием, что в этом случае символ a может быть последним символом предложения языка даже при r, отличном от $\varnothing$.

Таким образом, интерпретация правил (I), (2) включает как определение допустимого множества конкатенаций с данным терминальным символом правила, так и множества правил преемников либо определяет, что данный символ является конечным символом предложения языка. Грамматика языка задается четверкой:

$$G = (T, \mathfrak{A}, R, r_0) ,$$

где r_0 -аксиома грамматики, а каждому элементу из R поставлено в соответствие множество различных правил вида (I), (2).

Если задана такая грамматика, то для порождения необходимо из подмножества правил, соответствующих аксиоме r_0 , выбрать любое одно. Левый терм в этом правиле является первым символом порождаемого предложения, а имя r справа является именем подмножества правил-преемников. Из этого подмножества выбирается любое одно, левый терм в этом правиле является вторым символом порождаемого предложения и т.д.Если справа в текущем выбранном правиле стоит символ $\varnothing$, то порождение предложения закончено; если данное правило имеет W_S - отношение вида $\xrightarrow{W_S}$, то порождение предложения может быть закончено или

продолжено. Таким образом, конец порождения предложения определяется данным правилом, а его продолжение - правилом-преемником, переход к которому осуществляется согласно интерпретации правил, включающих надлежащее использование стеков.

Вместе с тем, приведенная грамматика весьма эффективна для целей распознавания и для использования в специализированных устройствах контроля и подготовки синтаксически правильных данных. Действительно,последовательность символов является предложением языка, если ее первый символ является символом,встречающимся в левой части одного из правил r_0 , каждый следующий - является символом, встречающимся в левой части одного из правил - преемников примененного правила, а последний символ встречается в одном из правил вида (2) или в правиле с символом $\varnothing$ в правой части.

Пример I. Грамматика $G = (T, \mathfrak{A}, R, r_0)$:

$T = \{a, b, c\}, \quad \mathfrak{A} = \{\varnothing, 1, 2\},$

$R = \{\varnothing, r_0, r_1, r_2, r_3\},$

$r_0 : \{a \xrightarrow{W_1(1,1)} r_1\},$

$r_1 : \{a \xrightarrow{W_1(2,2)} r_1,\ b \xrightarrow{W_2(2,\varnothing)} r_2,\ b \xrightarrow{W_2(1,\varnothing)} r_3\},$

$r_2 : \{b \xrightarrow{W_2(2,\varnothing)} r_2,\ b \xrightarrow{W_2(1,\varnothing)} r_3\},$

$r_3 : \{c \xrightarrow{W_2(\varnothing,2)} r_3,\ c \xrightarrow{W_2(\varnothing,1)} \varnothing\}$

порождает язык $L(G) = a^n b^n c^n$, где $n \geq 1$ - произвольное целое, обозначающее число повторений соответствующего символа.

Пример 2. Грамматика $G = (T, R, r_0)$:

$T = \{a, b, c\},$

$R = \{\varnothing, r_0, r_1, r_2\},$

$r_0 : \{a \xrightarrow{W_0} r_1,\ b \xrightarrow{W_0} \varnothing,\ a \xrightarrow{W_0} r_2\},$

$r_1 : \{b \xrightarrow{W_0} r_2,\ c \xrightarrow{W_0} \varnothing,\ b \xrightarrow{W_0} \varnothing\},$

$r_2 : \{b \xrightarrow{W_0} \varnothing,\ c \xrightarrow{W_0} r_1\}$

задает язык, грамматика которого в бэкусово-науровской форме I описывается следующими правилами:

$$\langle A\rangle ::= a\langle B\rangle \mid b$$
$$\langle B\rangle ::= c\langle c\rangle \mid \langle c\rangle \mid b$$
$$\langle C\rangle ::= b\langle D\rangle \mid c \mid b$$
$$\langle Д\rangle ::= c\langle c\rangle \mid b$$

Цепочка терминальных символов $\varphi=(a_1 a_2,\dots,a_k)$ является W - выводом φ (обозначение: $a\overset{W}{\Rightarrow}\varphi$), если существует последовательность правил-преемников, порождающих эту цепочку, начиная от аксиомы грамматики. Если при этом a_k порождается правилом вида (2) или (I) с символом $\varnothing_r$ в правой части, то W -вывод φ называется законченным и обозначается $a_1\overset{W}{\models}\varphi$. Язык, порождаемый R - грамматикой $G = (T, \mathfrak{A}, R, z_0)$:

$$L(G)=\{(a_1\dots a_k)\, a_i\in T,\ 1\le i\le k,\ a_1\overset{W}{\models} a_1\dots a_k\}.$$

Терминальные символы a_j и a_{j+k} $(k>0)$ в цепочке $\varphi=(a\dots \dots a_j\dots a_{j+k})$, порождаемые правилами с $W_1(\nu_1,\dots,\nu_n)$ и $W_2(\nu_1',\dots \nu_n')$, соответственно, называются базовым начальным и базовым конечным для i-го стека, если $\nu_i=\nu_i'$, $1\le i\le n$.

Правило R -грамматики называется фиктивным, если не существует законченного W - вывода φ, в котором данное правило применялось по крайней мере один раз. Грамматика, не содержащая фиктивных правил, называется приведенной.

R - грамматика является детерминированной, если терминальные символы в левых частях каждого подмножества правил совпадают лишь в правилах, содержащих W_2 с различными наборами аргументов.

П р и м е р 3. Язык, определяемый R - грамматикой второго примера, задается следующей приведенной детерминированной R - грамматикой $G=(T, R, z_0)$:

$$T=\{a,b,c\},\quad R=\{\varnothing, z_0, z_1, z_2, z_3\},$$
$$z_0:\{\xrightarrow{W_0} z_1,\quad b\xrightarrow{W_0}\varnothing\},$$
$$z_1:\{\xrightarrow{W_0} z_2,\quad c\xrightarrow{W_0} z_3\};$$
$$z_2:\{\xrightarrow{W_0}\varnothing,\quad c\xrightarrow{W_0} z_3\},$$
$$z_3:\{b\xrightarrow{W_0} z_2,\quad c\xrightarrow{W_0}\varnothing\}.$$

3°. Из множества R -грамматики выделим следующие, интересные для практических приложений классы:

I. Пусть каждое подмножество правил, соответствующих элементам множества имен R, содержит только одно правило. Грамматики данного вида назовем м о н а р н ы м и.

2. Пусть для всех правил грамматики $S = 0$. Такую грамматику назовем о д н о с т о р о н н е й л и н е й н о й.

3. Если в любом порождении слов все базовые конечные символы, соответствующие базовому начальному, порождаются правилами с совпадающими правыми частями, то такая грамматика называется л е в о к о н т е к с т н о й.

4. Если элементы множества R могут быть упорядочены так, что при любом порождении преемником может быть либо правило этого же подмножества правил, либо из любого следующего в упорядоченной последовательности подмножеств правил, то такая грамматика называется п о с л е д о в а т е л ь н о й.

Характерной особенностью монарной грамматики является то, что для них может быть построен универсальный алгоритм исправления одиночных ошибок. Хотя этими грамматиками порождается весьма узкий класс языков, последние находят широкое применение в качестве компонент, используемых на практике языков.

ТЕОРЕМА. О д н о с т о р о н н я я л и н е й н а я R- г р а м м а т и к а п р е д с т а в л я е т к о н е ч н ы й а в т о м а т в с м ы с л е Х о м с к о г о [2], и н а о б о р о т.

Устройство, реализующее подготовку синтаксически правильных программ и данных, описываемых односторонними линейными грамматиками, может не иметь стековой памяти.

Левоконтекстная R -грамматика это грамматика, в которой эффективно описывается довольно широкий класс практических языков. Например, язык АЛГОЛ - 60, из которого исключены логические выражения, может быть некоторым образом описан левоконтекстными R -грамматиками с помощью примерно I20 правил. Интересно отметить, что небольшие изменения в грамматике языка АЛГОЛ-60 переводят его в класс языков, описываемых левоконтекстными R -грамматиками. Для этого в грамматике языка [I], заданной в бекусово-науровской форме необходимо положить:

I. в разделе 4.2.I:

< оператор присваивания > ::= < список левой части > < А выражение > |< список левой части > < логическое выражение >

2. в разделе 3:

< выражение> ::= < А выражение > |< логическое выражение > |< именующее выражение >

3. в разделе 3.4.I:

< отношение> ::= < простое А выражение > < операция отношения > < простое арифметическое выражение >

4. Добавить раздел:

3.6. А выражение

3.6.I. – Синтаксис.

< первичное А выражение > ::= < число без знака > | < переменная > |< указатель функции > |< знак операции типа сложения >(< арифметическое выражение >)< А множитель > ::= < первичное А выражение > |< А множитель > ↑ < первичное выражение ><А терм > ::= < А множитель > | < А терм > < знак операции типа умножения > < множитель > < простое А выражение > ::= <А терм > |< знак операции типа сложения > <терм > |< простое А выражение > < знак операции типа сложения > < терм > < А выражение > ::= < простое А выражение >

Синтаксис <А выражения> отличается от синтаксиса < арифметического выражения > тем, что в нем запрещены <условные А выражения> и конструкции, начинающиеся с круглых скобок. В <А выражении > круглой скобке должен предшествовать < знак операции типа сложения >, а условные выражения допускаются только в круглых скобках. Например, конструкции вида((a +(c b... или вида $if\ a \neq$... должны быть заменены на +((a +(b ... и +($if\ a \neq$...) ...) соответственно.

Очевидно, язык с указанными изменениями и дополнениями является собственым подмножеством языка АЛГОЛ-60 и сами ограничения не уменьшают мощности языка.

Введение последовательной R -грамматики мотивировано той легкостью, с которой порождение в этой грамматике может быть осуществлено при помощи технического устройства. Если язык задан последовательной R-грамматикой, то соответствующее устройство подготовки синтаксически правильных программ и данных может не иметь долговременную память для хранения грамматики языка. Вместо долговременной памяти может быть использовано, например, технически более простое и дешевое устройство считывания с перфоленты.

Пусть $\lambda_0 = \{L / L$ – порождается монарной R -грамматикой $\}$, $\lambda_1 = \{L / L$ – порождается односторонней линейной R -грамматикой $\}$, $\lambda_2 = \{L / L$ – порождается левоконтекстной R – грамматикой $\}$, $\lambda_3 = \{L / L$ –порождается последовательной R – грамматикой $\}$.

С точки зрения порождающей способности эти системы соотносятся друг с другом следующим образом.

ТЕОРЕМА. $\lambda_0 \subset \lambda_1 \subset \lambda_2$, где оба включения собственные.

В доказательстве нуждается лишь собственный характер включений. Для этого построим язык $L_1 \in \lambda_1 \setminus \lambda_0$ и $L_2 \in \lambda_2 \setminus \lambda_1$. Нетрудно убедиться, что язык $a^n b^m c$ не порождается монарной и порождается следующей односторонней линейной R -грамматикой $G = (T, R, r_0)$:

$$T = \{a, b, c\}, \quad R = \{\varnothing, r_0, r_1\},$$

$$r_0 : \{a \xrightarrow{W_0} r_0, \quad b \xrightarrow{W_0} r_1, \ c \xrightarrow{W_0} \varnothing\},$$

$$r_1 : \{b \xrightarrow{W_0} r_1, \quad c \xrightarrow{W_0} \varnothing\}.$$

Известно [2], что язык $L_2 = a^n b^n c^m$ не порождается односторонней линейной грамматикой. Этот язык порождается следующей левоконтекстной R -грамматикой $G = (T, \mathfrak{A}, R, r_0)$:

$$T = \{a, b, c\}, \quad \mathfrak{A} = \{\varnothing, 1, 2\},$$

$$R = \{r_0, r_1, r_2, r_3\},$$

$$r_0 : \{a \xrightarrow{W_1(1)} r_1, \quad c \xrightarrow{W_0} r_3\},$$

$$r_1 : \{a \xrightarrow{W_1(2)} r_1, \ b \xrightarrow{W_2(2)} r_2, \ b \xrightarrow{W_2(1)} r_3\},$$

$$r_2 : \{b \xrightarrow{W_2(2)} r_2, \ b \xrightarrow{W_2(1)} r_3\},$$

$$r_3 : \{c \xrightarrow{W_0} r_3\}.$$

Очевидно, любую $\mathcal{R}$ -грамматику, содержащую правила вида [2], можно свести к эквивалентной ей (в смысле эквивалентности порождаемых языков) грамматике, содержащей только правила вида I . Для этого достаточно каждое правило вида $a \xrightarrow{W_g} z$ заменить парой правил:

$$a \xrightarrow{W_g} \phi$$
$$a \xrightarrow{W_g} z$$

Для языков $\{L_\alpha\}_{\alpha\in\Gamma}, \Gamma=\{1,\dots,N\}$, задаваемых $\mathcal{R}$ -грамматиками, не содержащими правил вида [2], справедливы следующие утверждения.

ТЕОРЕМА. Язык $L=\{L\}$ определяется $\mathcal{R}$ -грамматикой языка L, у которой все правила вида $a \xrightarrow{W_g} \phi$ заменены парами правил $a \xrightarrow{W_g} \phi$, $a \xrightarrow{W_g} z_o$.

ТЕОРЕМА. Язык $L=L_1 \cdot L_2 \cdot \ldots \cdot L_N$ определяется $\mathcal{R}$ -грамматикой $G=(T, \mathfrak{A}, \mathcal{R}, z_o)$, для которой $T=\bigcup_{\alpha\in\Gamma} T_\alpha, \mathfrak{A}=\bigcup_{\alpha\in\Gamma} \mathfrak{A}_\alpha, \mathcal{R}=\bigcup_{\alpha\in\Gamma} \mathcal{R}_\alpha, z_o=z_{o\alpha}$, и для грамматики $G_{\alpha-1}$ $(\alpha\in\Gamma)$ правила вида $a \xrightarrow{W_g} \phi$ заменены на $a \xrightarrow{W_g} z_{o\alpha}$.

ТЕОРЕМА. Язык $L=\bigcup_{\alpha\in\Gamma} L_\alpha$ определяется $\mathcal{R}$-грамматикой $G=(T, \mathfrak{A}, \mathcal{R}, z_o)$, у которой $T=\bigcup_{\alpha\in\Gamma} T_\alpha, \mathfrak{A}=\bigcup_{\alpha\in\Gamma} \mathfrak{A}_\alpha, \mathcal{R}=\bigcup_{\alpha\in\Gamma} \mathcal{R}\setminus z_{o\alpha}$, z_o - имя теоретико-множественного объединения правил, соответствующих всем $z_{o\alpha}$.

В последних двух утверждениях W_g - отношения в результирующих грамматиках являются функциями, аргументы которых представляют собой упорядоченную последовательность аргументов всех функций W_g^α $(\alpha\in\Gamma)$.

При интерпретации правил результирующей грамматики это соответствует использованию всех стеков языков компонент.

IУ. $\mathcal{R}$ -грамматики практических языков содержат большое число синтаксически эквивалентных термов.

ОПРЕДЕЛЕНИЕ. Два терминальных элемента $a,b\in T$ называются синтаксически эквивалентными (обозначение $a \cong b$), если для любых терминальных цепочек ψ_1 , ψ_2 языков L из $\psi_1 a \psi_2 \in L$ следует $\psi_1 b \psi_2 \in L$, и наоборот.

Например, символы +,- в языке АЛГОЛ-60 синтаксически эквивалентны. Аналогично, символы цифр; букв, true, false и т. д. попарно синтаксически эквивалентны между собой.

Очевидно, что установить синтаксическую эквивалентность двух терминальных символов практического языка можно лишь с помощью его грамматики.

ТЕОРЕМА. В $\mathcal{R}$ - грамматике, если для любого подмножества правил, в котором используется терминальный символ b, имеется правило с терминальным символом a в левой части (и наоборот) и эти правила отличаются только левыми частями, то $a \cong b$.

Разобьем множество терминальных элементов на классы синтаксически эквивалентных элементов. Подмножество терминальных элементов, попарно различные элементы которого синтаксически эквивалентны, называется классом синтаксически эквивалентных элементов. Так, подмножество терминальных элементов $\{<, \geqslant, =, \neq, \leqslant, >\}$ языка АЛГОЛ-60 образует класс синтаксически эквивалентных элементов. Выберем из каждого класса синтаксически эквивалентных элементов по одному представителю, с которым в дальнейшем будем отождествлять любой элемент из данного класса. В качестве представителя каждого класса синтаксически эквивалентных элементов может быть взят любой символ, отличный от символов словаря T . Назовем символ, обозначающий класс синтаксически эквивалентных элементов, синтермом языка.

На практике удобно определять синтерм как некоторую функцию: $b=F(a,\tau)$,где b -синтерм, $a\in T$, τ - параметр соответствующего элемента множества имен $\mathcal{R}$ -грамматики языка. Таким образом, грамматика G в предлагаемом метаязыке:

$$G=(T, \mathfrak{A}, \mathcal{R}, z_o, \mathfrak{M}, F),$$

где $T=\{a_i\}_{i\in\Gamma_1}$ - терминальный словарь; $\Gamma_1=\{1,\dots, t\}$;

$\mathfrak{A}=\{\phi,\nu_i\}_{i\in\Gamma_2}$ – некоторый вспомогательный словарь, $\Gamma_2=\{1,\dots,n\}$; $\mathcal{R}=\{\tau_i^{\tau}\}_{i\in\Gamma,\tau\in\Gamma_4}$ – множество имен некоторых конечных подмножеств (возможно, пустых) правил вида $б\xrightarrow{W_s}\tau_i^{\tau}$ или $б\xrightarrow{W_s}.\ \tau_i^{\tau}$, где $б$ – синтерм; $\tau\in\mathcal{R}$; имя пустого подмножества правил – $\varnothing$; τ – некоторый параметр для определения синтермов правил преемников; $\xrightarrow{W_s}$ или $\xrightarrow{W_s}.$, $W_s=W_s(\nu_1,\nu_2,\dots,\nu_n)$, $\nu_i\in\mathfrak{A}$, $1\leq i\leq n$ – многомостное отношение, означающее, что синтерм слева от него некоторым образом конкатенирует с синтермом левой части любого правила из подмножества, имя которого указано справа от данного W_s – отношения; конкатенация синтермов определяется типом s (s = 0,I,2) W_s – отношения аналогично вышеописанному:

τ_0 – аксиома грамматики ;

$\mathfrak{M}=\{\tau_i\}_{i\in\Gamma_2}$ – множество параметров имен из $\mathcal{R}$;

F – функция, которая каждому элементу множества T и параметру $\tau\in\mathfrak{M}$ элемента множества имен $\mathcal{R}$ грамматики ставит в соответствие синтерм.

П р и м е р . Простое арифметическое выражение языка АЛГОЛ-60 [I], в котором

<первичное выражение> ::= <целое> | < простая переменная> | <простое арифметическое выражение> задается следующей $\mathcal{R}$-грамматикой:

$$G=(T,\ \mathfrak{A},\ \mathcal{R},\ \tau_0,\ \mathfrak{M},\ F),$$

$$T=\{a,b,\dots,y,z,A,B,\dots,Y,Z,0,1,\dots,9,+,-,\times,/,\div,\uparrow,),(\},$$

$$\mathfrak{A}=\{\phi,1,2\},$$

$\mathfrak{M}=\{1,2,3\}$, F – функция, задаваемая таблицей вида:

τ \ T	a	b	…	y	z	A	B	…	Y	Z	0	I	…	9	+	−	×	/	÷	↑	)	(
I	б	б	…	б	б	б	б	…	б	б	ц	ц	…	ц	+	+	×	×	×	×	)	(
2	б	б	…	б	б	б	б	…	б	б	ц	ц	…	ц	×	×	×	×	×	×	)	(
3	бц	бц	…	бц	бц	бц	бц	…	бц	бц	бц	бц	…	бц	×	×	×	×	×	×	)	(

$$\mathcal{R}=\{\tau_0^1,\tau_1^3,\tau_2^2,\tau_3^2,\tau_4^1,\tau_5^3,\tau_6^2,\tau_7^2,\tau_8^2,\tau_9^2\},$$

$$\tau_0^1:\{б\xrightarrow{W_0}\tau_1^3,\ ц\xrightarrow{W_0}\tau_2^2,\ +\xrightarrow{W_0}\tau_3^2,\ (\xrightarrow{W_1(1)}\tau_4^1\},$$

$$\tau_1^3:\{бц\xrightarrow{W_0}\tau_1^3,\ \times\xrightarrow{W_0}\tau_3^2\},$$

$$\tau_2^2:\{ц\xrightarrow{W_0}\tau_2^2,\ \times\xrightarrow{W_0}\tau_3^2\},$$

$$\tau_3^2:\{б\xrightarrow{W_0}\tau_1^3,\ ц\xrightarrow{W_0}\tau_2^2,\ (\xrightarrow{W_1(1)}\tau_4^1\},$$

$$\tau_4^1:\{б\xrightarrow{W_0}\tau_5^3,\ ц\xrightarrow{W_0}\tau_6^2,\ +\xrightarrow{W_0}\tau_7^2,\ (\xrightarrow{W_1(2)}\tau_4^1\},$$

$$\tau_5^3:\{бц\xrightarrow{W_0}\tau_5^3,\ \times\xrightarrow{W_0}\tau_7^2,\ )\xrightarrow{W_2(1)}\tau_9^2,\ (\xrightarrow{W_2(2)}\tau_8^2\},$$

$$\tau_6^2:\{ц\xrightarrow{W_0}\tau_6^2,\ \times\xrightarrow{W_0}\tau_7^2,\},$$

$$\tau_7^2:\{б\xrightarrow{W_0}\tau_5^3,\ ц\xrightarrow{W_0}\tau_6^2,\ (\xrightarrow{W_1(2)}\tau_4^1\},$$

$$\tau_8^2:\{\times\xrightarrow{W_0}\tau_7^2,\ )\longrightarrow\tau_9^2,\ )\xrightarrow{W_2(2)}\tau_8^2\},$$

$$\tau_9^2:\{\times\xrightarrow{W_0}\tau_3^2\}.$$

Использование синтерминального словаря, задаваемого некоторой функцией F, позволяет на практике существенно сократить исходную запись грамматики языка, а следовательно, упростить соответствующие алгоритмы контроля и схемы устройства подготовки синтаксически правильных программ и данных.

Л и т е р а т у р а

I. NAUR P.(Ed.). Revised report on the Algorithmic language ALGOL-60, Comm. ACM, vol.6. № I, 1963, p.I-I7 (Русский перевод: П.Наур (ред.)."Алгоритмический язык АЛГОЛ-60", Пересмотренное сообщение, М., "Мир", 1965).

2. CHOMSKY N. Formal properties of grammers, Handbook of Mathematical Psychology, v.2, ch.12, Wiley , 1963, p328-4I8 (Русский перевод: Н.Хомский, Формальные свойства грамматик, -"Кибернетический сборник", М., "Мир", 1966, вып.2, стр.I2I-230).