оптимизированном базисе. Известия АН СССР. Серия физики, 1975, № 3, с. 543-549.
4. О д е н Дж. Конечные элементы в нелинейной механике сплошных сред. - М.: Мир, 1976.
5. Теоретические основы и конструирование вычислительных алгоритмов математической физики. / Под ред. К.И.Бабенко. - М.: Наука, 1979.
6. С е г е р л и н д Дж. Применение метода конечных элементов. - М.: Мир, 1977.
7. К р а с н о п о л ь с к и й В.М. Автореферат кандидатской диссертации.- М.: 1978.
8. О р т е г а Дж., Р е й н б о л д т В. Итерационные методы решения нелинейных уравнений со многими неизвестными. - М.: Мир, 1975.
9. Lentini M., Pereyra V.A Variable Ordez Finite Difference Method for Nonlinear Multipoint Boundary Value Problems. Mathema- tics of Computation, 1974, vol. 28, No. 128, pp. 981-1003.

УДК 681.3.06.51

П.В.Гамин, В.В.Куликов

ГЕНЕРАЦИЯ ПРОГРАММ СИНТАКСИЧЕСКОГО АНАЛИЗАТОРА

Транслирующие компилирующие и интерпретирующие системы имеют большое практическое значение при разработке программного обеспечения систем реального времени информационно-вычислительных комплексов (СПО СРВ ИВК) на базе мини-ЭВМ. Широкое внедрение этих ЭВМ в практику научных исследований делает актуальной задачу развития транслирующих систем для реализации языковых средств общения экспериментатора-пользователя ИВК. К таким средствам можно отнести диалоговые проблемно-ориентированные языки, языки подготовки данных, языки запросов информационно-поисковых систем общего и специального назначения.

Общая тенденция автоматизации проектирования в технике нашла свое отражение и в задачах автоматизации транслирующих систем. В

данной работе рассматривается метод, использованный для разработки программной системы автоматического построения синтаксически-ориентированного анализатора. Эта программная система используется для реализации диалогового интерпретатора и программных средств подготовки исходных данных автоматизированной системы управления экспериментальной технологической установкой на базе ЭВМ М-6000 с операционной системой ДОС РВ. Система генерации программ синтаксически управляемого анализатора используется в составе ДОС ЕС.

Процесс анализа исходной строки символов некоторого формального языка $L(G)$ заключается в определении последовательности применений правил вывода $\rho_i \in P$ грамматики $G=(V_N, V_T, S, P)$, где V_N - нетерминальный алфавит; V_T - терминальный алфавит; $V_N \cap V_T = \phi$; P - множество правил вывода $\rho_i : \psi_i \Rightarrow \varphi_{ij} / i = \overline{1,n}$; $j = \overline{1,m_i}$; $S \in V_N$ - начальный символ грамматики.

Строки x, выводимые из начального символа S за конечное число применений правил вывода $\rho_i \in P$, образуют множество $L(G) = \{x / x \in V_T^* \wedge S \Rightarrow x\}$ допустимых строк языка. Здесь V_T^* - множество всех строк (включая пустой символ e) над алфавитом V_T. Особый интерес в нашем случае представляют однозначные грамматики, т.е. грамматики G, у которых для каждой допустимой строки x, выводимой из символа S, все ее выводы совпадают. Дополнительно введем ограничение на левые части правил вывода ψ_i, полагая $|\psi_i| = 1$, т.е. потребуем, чтобы грамматика G была контекстно-свободной.

В грамматиках не фиксируется порядок применения правил вывода, т.е. они не являются алгоритмами и поэтому вводятся канонические формы вывода/распознавания. Наиболее естественную каноническую форму можно получить, если учесть естественную для правил вывода КС-грамматик подчиненность нетерминальных символов, например, используя алгоритм задания частичного порядка на множестве P [4].

Определение правила вывода $(\rho_i : \psi_i \Rightarrow \varphi_{ij}) \in P$ сводится к распознаванию всех символов цепочки φ_{ij} в алфавите $V = V_T \cup V_N$. При этом символы этой цепочки определяют допустимые переходы в синтаксическом графе и распознавание использованного правила вывода ρ_i. Символы $A_i \in V$, по которым возможен переход по дуге синтаксического графа, называют допустимыми. По допустимым символам можно построить синтаксический анализатор в форме магазинного

автомата (МП - автомата [1, 2]), являющегося в отличие от грамматики (как формальной порождающей системы) алгоритмом. При этом для однозначной КС-грамматики будет построен детерминированный МП-автомат. Можно отметить, что при таком подходе грамматика $G(V_N, V_T, S, P)$ является вспомогательной формальной системой, необходимой для построения состояний МП-автомата.

Множество состояний МП-автомата строится, исходя из понятия конфигураций, получаемых из правил $(p_i : \psi_i \Rightarrow \varphi_{ij} / i = \overline{1,n}\ ;\ j = \overline{1, m_i}) \in P$ следующим образом. Начальная конфигурация K_0^S образуется из правил вида $\{S \Rightarrow_0 \varphi_{Si} = A_j \psi\ ;\ j = \overline{1, m_S}\ ;\ A_j \in V;\ \psi \in V^*\}$ добавлением метасимвола (маркера) $\circ \notin V$ перед первым символом подстроки φ_{Sj}. Символы $A_j \in V$, отмеченные маркером, являются символами перехода по синтаксическому графу. При этом символы $A_i \notin V_N$ входят в левые части правил вывода, т.е. $\psi_i = A_i$. Множество правил вывода $(A_i \Rightarrow_0 \varphi) \in P$ совместно с начальной конфигурацией K_0^S образуют полную группу конфигураций K^S.

Символы перехода A_j образуют начальные конфигурации $K_0^{A_j}$ вида $\{A_i \Rightarrow x_0 A_{j0} \psi / x,\ \psi \in V^*,\ A_i \in V_a,\ A_j \in V\}$. Расширение начальной конфигурации до полной группы производится путем добавления правил вывода $(B_K \Rightarrow_0 \varphi) \in P$, где символы перехода B_K являются первыми символами подстроки $\varphi \in V$.

Конфигурации вида $\{\psi_i \Rightarrow \varphi_{ij0}\}$ не имеют символов перехода в другие состояния, они указывают на необходимость замены (редукции) подстроки φ_{ij} на подстроку (символ $\psi A_i \in V_N$) ψ_i. Для однозначной КС-грамматики полная группа конфигураций должна содержать не более одной заключительной конфигурации.

Описанный процесс для однозначных КС-грамматик приводит к получению конечного числа групп конфигураций. Группы символов перехода совместно с номерами состояний, в которые осуществляется переход по очередному символу, образуют состояния МП-автомата:

$$A = (Q, V, \Gamma, \delta, q_0, z_0, F),$$

где $Q = \bar{Q} \cup \hat{Q}$ - конечное множество состояний;

$\bar{Q}$ - множество состояний, из которых возможен переход по очередному символу $A_j \in V_T \cup V_N$ грамматики $G = (V_N, V_T, S, P)$;

$\hat{Q}$ - множество состояний, в которых производится редукция φ_{ij} на ψ_i ;

V - конечный входной алфавит;

Γ - конечный алфавит магазинных символов;

$\delta: (Q \times V \times \Gamma) \Rightarrow P(Q \times \Gamma^*)$;

$q_0 \in \bar{Q}$ - начальное состояние МП-автомата;

$z_0 \in \Gamma$ - начальный символ магазина;

$F \in Q$ - множество конечных состояний МП-автомата.

Множество $Q \times V \times \Gamma$ выполняет функцию программы МП-автомата, номера состояний образуют множество Γ . Программа МП-автомата реализуется тремя типами команд (множество Y):

чтение со входной ленты очередного символа, изменение состояния магазина;

распознавание примененного правила вывода $p_i : \psi_i \Rightarrow \varphi_{ij} \in P$;

редуцирование строки φ_{ij} к строке ψ_i '.

На этапе оптимизации программы МП-автомата могут быть включены команды безусловного перехода.

Рассмотрим программную реализацию описанной процедуры построения множества состояний МП-автомата для случая, когда команды МП-автомата оформляются в виде макрокоманд языка АССЕМБЛЕР ДОС ЕС

Система генерации множества состояний МП-автомата. Система генерации множества состояний МП-автомата реализована на языке высокого уровня $PL/1$ и состоит из двух подсистем (рис. I), выполняемых под управлением операционной системы ДОС ЕС на ЭВМ класса РЯД.

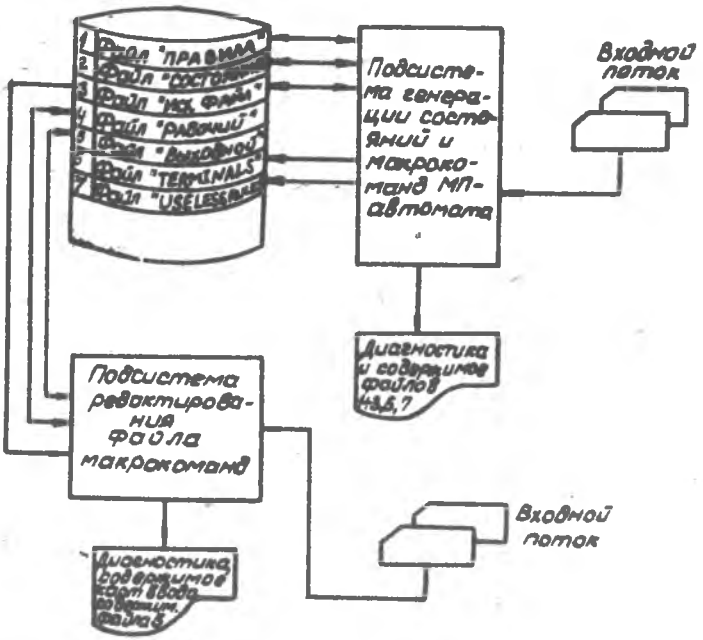
Рассмотрим каждую подсистему в отдельности.

Подсистема генерации собственно состояний МП-автомата. Подсистема генерации собственно состояний МП-автомата ПГС реализована в виде оверлейной программы, состоящей из следующих фаз:

I. Фаза $SVETA$ является корневой и служит для управления всей подсистемой генерации.

2. Фаза $GENERS$ - обеспечивает ввод правил исходной грамматики, проверку правил на синтаксис и генерацию состояний МП-автомата.

3. Фаза $MACROS$ служит для генерации собственно макрокоманд синтаксического анализатора в формате макрокоманд языка АССЕМБЛЕР ДОС ЕС.



Р и с. I. Схема работы и состав системы генерации программ синтаксического анализатора

4. Фаза *OPTIMS* служит для оптимизации сгенерированного множества состояний МП-автомата (например, для исключения дублирующих редукций и т.д.).

5. Фаза *STATIS* служит для сбора вспомогательной информации о рассматриваемой грамматике:

I) построение алфавита нетерминальных символов;

2) построение алфавита терминальных символов;

3) анализ семантики грамматики в смысле избыточности правил, входящих во множество правил P .

Данная подсистема работает с информацией, которая описывает исследуемую грамматику G . Эта информация по существу представляет множества V_N, V_T, S, P, т.е. программа ввода вводит множество правил подстановки (PUS), а фаза 5 определяет по нему множества V_N и V_T

Формат записи правил подстановки на перфокартах произвольный и имеет вид

$\psi_i \Rightarrow \varphi_{ij} \equiv$ #<имя><имя><имя> ... <имя>

Всего в правиле должно быть не более 10 имен. Запрещается использовать в имени символы: #, <, >.

Знак # идентифицирует начало нового правила.

Символ $S \in V_T$ записывается на отдельной перфокарте и размещается в позициях 1-10.

Например <ДАТА>

ПГС оперирует с пятью файлами, имеющими прямую организацию. Это следующие по назначению файлы:

1. Файл правила вывода (P) "ПРАВИЛА".
2. Файл терминальных символов (V_T) „TERMINALS".
3. Файл состояний МП-автомата (F) "СОСТОЯНИЯ".
4. Файл правил, являющихся избыточными, т.е. не участвующими в генерации состояний МП-автомата „USELESSRULES".
5. Файл собственно макрокоманд синтаксического анализатора "ИСХ. ФАЙЛ".

Каждый модуль подсистемы выдает диагностическую информацию о завершении работы модуля (т.е. генерирует код завершения). Код завершения представляет собой символьную константу длиной 12 байт, которая содержит следующую информацию:

1) первые три символа - номер кода;

2) следующие 6 символов - имя модуля, выдавшего код;

3) следующие 3 символа - дополнительная информация, учитывающая специфику модуля, например, номер карты входного потока, в которой содержится ошибка.

В процессе работы ПГС формируются файлы 1-5, диагностический вывод и вывод на системное устройство, отражающий содержание сформированных файлов.

Файл макрокоманд 5 содержит макрокоманды вида

CAB T1 = имя, MET1 = метка

RDN VAR = имя, NUM = число (1)

где имя - символическое имя 1-8 символов;
число - число символов редукции;
метка - метка перехода.

Если имеется синтаксический анализатор, использующий другой формат макрокоманд, то путем простых изменений фазы можно сформировать генерацию нужного формата макрокоманд.

Подсистема редактирования файла макрокоманд. Подсистема редактирования файла макрокоманд ПРФМ служит для добавления, удаления и изменения записей файла макрокоманд.

Необходимость создания данной подсистемы вытекает из того, что ПГС не в состоянии оформить до конца ассемблерную программу на основании файла макрокоманд для синтаксического анализатора.

Оформлением и логическим завершением построения программы анализа должен заниматься человек, он также включает построенную программу в библиотеку программ синтаксического анализа, если таковая существует.

Программист пишет пакет редактирования, который по исходному файлу 5 формирует выходной файл, который может быть назначен как входной поток обрабатывающей программе АССЕМБЛЕРа. ПРФМ состоит из одной программы *@EDITO* и оперирует с тремя файлами, имеющими прямую организацию:

1) исходный файл макрокоманд;

2) рабочий файл;

3) выходной файл;

Подсистема редактирования файла макрокоманд выдает на системный вывод *SYSLST* следующие данные:

входной поток - карты добавления или замены;

все записи файла макрокоманд после его редактирования;

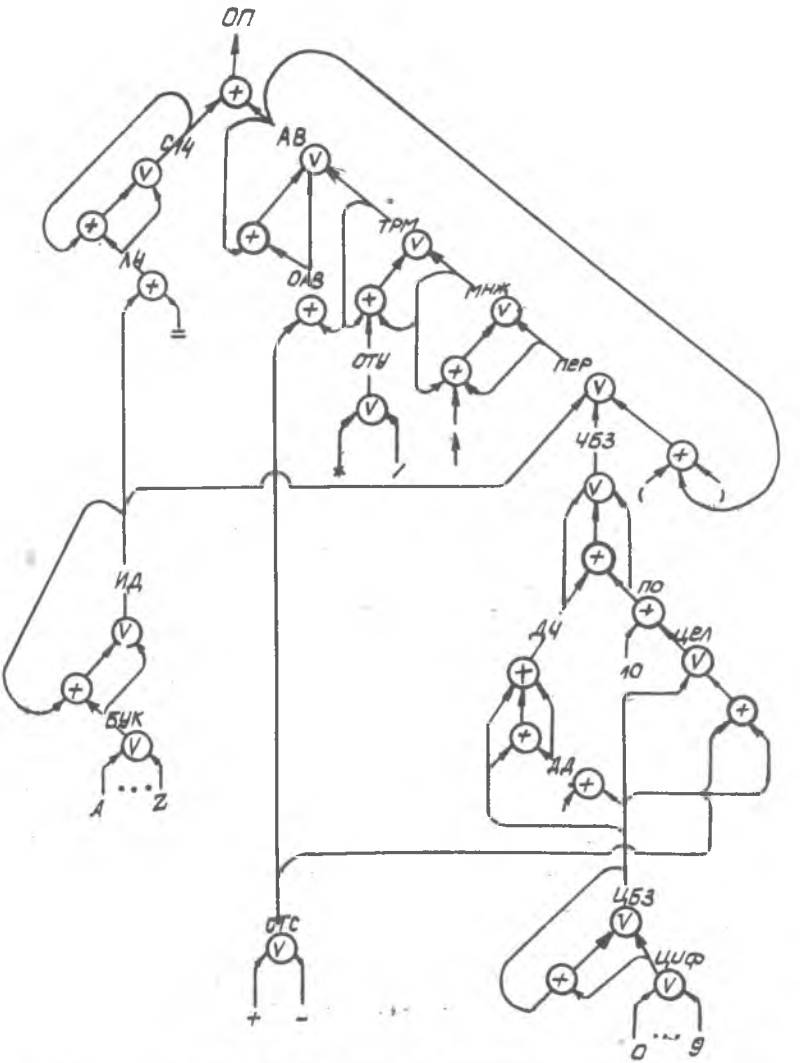
диагностические сообщения.

Диагностические сообщения представляют собой код завершения работы подсистемы и имеют структуру такую же, как и код завершения подсистемы ПГС.

ПРФМ может редактировать любой файл, имеющий организацию *REGIONAL* (I) и длину записей 80 байт [5 , 6].

Рассмотрим генерацию программы МП-автомата на примере грамматики простых арифметических выражений (ПАВ). Синтаксический граф ПАВ показан на рис. 2.

Для системы генерации грамматика задается множеством правил вывода P и начальным символом S . В нашем случае это множество будет иметь вид (расписано в формате ввода в ЭВМ):



Р и с. 2. Синтаксический граф грамматики простых арифметических выражений

Правило вывода	Номер правила
# < ОП > < СЛЧ > < АВ >	I
# < СЛЧ > < СЛЧ > < АЧ >	2
# < СЛЧ > < ЛЧ >	3
# < АВ > < АВ > < ОАВ >	4
# < АВ > < ТРМ >	5
# < АВ > < ОАВ >	6
# < ЛЧ > < ИД > < = >	7
# < ОАВ > < ОТС > < ТРМ >	8
# < ТРМ > < ТРМ > < ОТУ > < МНЖ >	9
# < ТРМ > < МНЖ >	I0
# < МНЖ > < МНЖ > < I > < ПЕР >	II
# < МНЖ > < ПЕР >	I2
# < ОТУ > < Ж >	I3
# < ОТУ > < / >	I4
# < ПЕР > < ЧБЗ >	I5
# < ПЕР > < ИД >	I6
# < ПЕР > < '(' > < АВ > < ')' >	I7
# < ИД > < ИД > < БУК >	I8
# < ЧБЗ > < ДЧ > < ПО >	I9
# < ЧБЗ > < ДЧ >	20
# < ЧБЗ > < ПО >	2I
# < ДЧ > < ЦБЗ > < ДД >	22
# < ДЧ > < ЦБЗ >	23
# < ДЧ > < ДД >	24
# < БУК > < А >	25
# < ПО > < Е > < ЦЕЛ >	26
# < ДД > < . > < ЦБЗ >	27
# < ЦЕЛ > < ОТС > < ЦБЗ >	28
# < ЦЕЛ > < ЦБЗ >	29
# < ЦБЗ > < ЦБЗ > < ЦИФ >	30
# < ЦБЗ > < ЦИФ >	3I
# < ОТС > < + >	32
# < ОТС > < - >	33
# < ЦИФ > < 9 >	34
# < ИД > < БУК >	35

Символы имеют следующее смысловое значение:

ОП - оператор присваивания начального символа,
СЛЧ - список левой части,
АВ - арифметическое выражение,
ЛЧ - левая часть,
ОАВ - одноместное арифметическое выражение,
ТРМ - терм,
МНТ - множитель,
ОТУ - операция типа умножения,
ПЕР - переменная,
ИД - идентификатор,
ЧБЗ - число без знака,
ДЧ - десятичное число,
БУК - буква,
ПО - порядок,
ДД - десятичная дробь,
ЦЕЛ - целое,
ЦБЗ - целое без знака,
ОТС - операция типа сложения,
ЦИФ - цифра,
= - знак "Присвоить",
I - знак "Возвести в степень",
* - знак "Умножить",
/ - знак "Делить",
(- знак "Левая скобка",
A - любая буква алфавита,
E - признак десятичного порядка,
. - разделитель дробной и целой части,
+ - знак "Сложить",
- - знак "Вычесть",
9 - любая цифра,
) - правая скобка.

Система генерации формирует файл терминальных символов, словарь терминальных символов и выдает на системный вывод в виде

```
* * * * * * * * * * * * * * * * * * * * * * * * * * *
*  Т Е Р М И Н А Л Ь Н Ы Е   С И М В О Л Ы     *
* * * * * * * * * * * * * * * * * * * * * * * * * * *
  ЧИСЛО ТЕРМИНАЛЬНЫХ СИМВОЛОВ = 11
```

TERM = (< = > , < I > , < * > , < / > , < (> , < A > ,
< E > , < . > , < + > , < - > , < 9 >).

Формирует алфавит грамматики (или список нетерминальных символов) и выдает на системное устройство вывода в виде

```
* * * * * * * * * * * * * * * * * * * * * * *
*     В Х О Д Н О Й   А Л Ф А В И Т         *
* * * * * * * * * * * * * * * * * * * * * * *
```

T = (< ОП > , < СЛЧ > , < АВ > , < ЛЧ > , < ОАВ > , < ТРМ > ,
< МНЖ > , < ОТУ > , < ПЕР > , < ИД > , < ЧБЗ > , < ДЧ >
< БУК > , < ПО > , < ДД > , < ЦЕЛ > , < ЦБЗ > , < ОТС >
< ЦИФ >)

ЧИСЛО СИМВОЛОВ В АЛФАВИТЕ = 19.

Далее система генерации программы МП-автомата выдает список правил вывода в форме строк

```
  * * *  . . . . . . . . . . . . . . . . . . * *
I *  < ОП >  =  < СЛЧ > < АВ >                 *
. *          .                                 *
. .          .                                 .
. .          .                                 .

. .          .                                 .
. .          .                                 .
35 *   < ИД > =   < БУК >                       *
  * . . . . . . . . . . . . . . . . . . . . * *
```

Состояния МП-автомата генерируются системой в виде следующих строк: XXX YYY ZZ <@@@@@@@@> SSSβ

где XXX - метка состояния;

YYY - номер правила;

ZZ - положение маркера 0 ;

@@@@@@@@ - символ перехода или символ сравнения;

SSS - метка перехода в следующее состояние МП-автомата;

$\beta = \left\{ \begin{matrix} * \\ u \end{matrix} \right.$ - пробел $\}$.

β = * - означает, что строка, характеризующая состояние МП-автомата, является избыточной.

Параметр β используется для оптимального построения программы МП-автомата.

Если система завершила генерацию очередного состояния, то она сообщает об этом устройстве системного вывода:

ЗАВЕРШЕНА ГЕНЕРАЦИЯ СОСТОЯНИЯ N = XXX

В рассматриваемом примере число сгенерированных состояний равно 46.

После завершения генерации всех состояний МП-автомата и оптимизации этого множества система генерирует макрокоманды типа (I). Получив листинг содержимого файла макрокоманд, программист может отредактировать его с помощью подсистемы ПРФМ в соответствии со своими целями и, запустив этот отредактированный модуль, получить синтаксический анализатор (ПАВ в данном примере).

Выводы

1. Система генерации позволяет создавать программы синтаксических анализаторов на основе множества правил вывода некоторой КС-грамматики G. Причем, она обеспечивает программисту контроль множества P и сбор вспомогательных сведений об используемой грамматике: V_N, V_T, число избыточных правил и их номера.

2. В соответствии с п. I существенно облегчается построение различных синтаксически-ориентированных анализаторов.

Литература

I. Педанов И.Е., Шумей А.С. О расширении класса грамматик $LR[0]$. Автоматика и телемеханика, I973, № II, с. I50-I53.

2. Шумей А.С., Зонис В.С. О синтаксическом анализе по однозначным грамматикам. Программирование, I975, № 3.

3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. Синтаксический анализ. / Пер. с англ./ - М.: Мир, I978, т. I.

4. Ингерман П. Синтаксически ориентированный транслятор. - М.: Мир, I969.

5. Шатровский Л.И. Макропрограммирование в вычислительной среде ДОС/ЕС. - М.: Советское радио, 1979.
6. Инженерные расчеты на ЭВМ. Справочное пособие под редакцией проф., д-ра физ.-мат. наук Троицкого В.А. - Л.: Машиностроение, 1979.

УДК 681.3.06:5

В.В.Куликов, М.А.Шамашов

АВТОМАТИЗАЦИЯ ПРОЕКТИРОВАНИЯ СИНТАКСИЧЕСКИХ АНАЛИЗАТОРОВ ПРОБЛЕМНО-ОРИЕНТИРОВАННЫХ ЯЗЫКОВ СИСТЕМ АВТОМАТИЗАЦИИ ЭКСПЕРИМЕНТА

Решение задач автоматизации экспериментальных исследований, как правило, требует использования специально разрабатываемого проблемно-ориентированного программного обеспечения (ПО). В частности, возникает необходимость построения трансляторов для языков управления экспериментом, реализуемых в диалоговом и пакетном режимах, и систем подготовки информационных массивов заданной структуры [1-3]. Подобные задачи возникают и при построении моделей специализированных УВМ, входящих в состав систем автоматизации эксперимента [4, 5]. Большая стоимость ПО требует создания инструментальных систем автоматизации программирования (ИСАП).

В предлагаемой статье рассматривается один из подходов автоматизации процесса построения синтаксических анализаторов для трансляторов (САТ) проблемно-ориентированных языков - одной из подсистем ИСАП.

Разработанная система автоматизации проектирования САТ базируется на уже известном наборе команд магазинного преобразователя (МП-преобразователя) [6]. Исходная КС-грамматика задана в виде пятерки $G=(V_N, V_T, S, P, \#)$, где V_T и V_N - конечные алфавиты основных и вспомогательных символов, $V_N \cap V_T = O$, S - начальный символ грамматики, $S \in V_N$, P - множество правил вывода вида

$$R_i(\psi_i \to \varphi_{ij} \mid i=\overline{1,n};\ j=\overline{1,m_i};\ |\psi_i|=1) \in P.$$