Тема 1. Синтаксически управляемые определения

Синтаксически управляемое определение (СУО) представляет собой контекстно-свободную грамматику, дополненную *атрибутами* и *семантическими правилами*. Атрибуты связываются с символами грамматики (терминалами и нетерминалами) и могут иметь любой вид: число, тип данных, строку, ссылку на запись таблицы символов, адрес памяти и т.п. Атрибут a символа X грамматики обычно записывается в виде $X.a$. Семантические правила связываются с продукциями грамматики и описывают способ вычисления атрибутов. Обычно семантические правила записывают в виде $b := f(c_1, c_2, \ldots, c_k)$, где f – некоторая функция, записываемая часто как выражение; $b, c_1, c_2, \ldots, c_k$ – атрибуты, что говорит о том, что значение атрибута b зависит от значений атрибутов $c_1, c_2, \ldots, c_k$.

Если продукция в СУО содержит рекурсию, обычно нетерминал в левой части продукции записывают без индекса, а в правой части этот же нетерминал записывают с индексом, например, $R \to +TR_1$. Аналогично, если имеется несколько вхождений одного и того же символа грамматики в правую часть продукции, они записываются с различными индексами, например, $E \to E_1 + E_2$. Это связано с тем, что в дереве разбора таким символом грамматики помечены разные вершины, в которых один и тот же атрибут может иметь различные значения.

1.1. Типы атрибутов

Для нетерминалов выделяют два типа атрибутов: синтезируемые и наследуемые.

Синтезируемый атрибут определяется связанным с продукцией семантическим правилом для нетерминала A в левой части продукции. Значение синтезируемого атрибута вычисляется только с использованием атрибутов символов правой части продукции и атрибутов нетерминала A. Другими словами, если в дереве разбора нетерминалу A соответствует узел N, значение синтезируемого атрибута $A.s$ в узле N определяется только с использованием значений атрибутов в дочерних по отношению к N узлах и в самом узле N.

Наследуемый атрибут определяется связанным с продукцией семантическим правилом для нетерминала B в правой части продукции. Значение наследуемого атрибута вычисляется только с использованием атрибутов нетерминала A в левой части продукции и атрибутов символов правой части продукции (включая и атрибуты нетерминала B). Другими словами, если в дереве разбора нетерминалу B соответствует узел M, значение наследуемого атрибута $B.i$ в узле M определяется с использованием значений атрибутов в родительском по отношению к M узле, в самом узле M и братьях узла M.

Терминал может иметь только синтезируемый атрибут, который является атрибутом токена, передаваемого лексическим анализатором. Поэтому в СУО не должно быть семантических правил для вычисления значений атрибутов терминалов.

Иногда семантическое правило записывается как вызов процедуры или выполнение фрагмента программы (в таких случаях говорят, что правило имеет побочное действие). Побочное действие называется *контролируемым*, если оно не изменяет порядок вычисления атрибутов и не препятствует их вычислению. Правила с контролируемыми побочными действиями можно рассматривать как правила, определяющие значения фиктивных синтезируемых атрибутов нетерминала в левой части связанной продукции. СУО, в котором семантические правила не имеют побочных действий, называется *атрибутной* грамматикой.

СУО, которые включают только синтезируемые атрибуты, называются *S-атрибутными* СУО. В таком СУО семантическое правило вычисляет значение синтезируемого атрибута нетерминала в левой части продукции, используя значения синтезируемых атрибутов символов грамматики в правой части продукции.

Пример *S*-атрибутного СУО вычисления значения арифметического выражения представлен в табл. 1.

Таблица 1

S-атрибутное СУО вычисления арифметического выражения

Продукция	Семантические правила
1) $S \to E\perp$	$S.val := E.val$
2) $E \to E_1 + T$	$E.val := E_1.val + T.val$
3) $E \to T$	$E.val := T.val$
4) $T \to T_1 * F$	$T.val := T_1.val * F.val$
5) $T \to F$	$T.val := F.val$
6) $F \to (E)$	$F.val := E.val$
7) $F \to$ **num**	$F.val := GetVal($**num**$.ns)$

Каждый нетерминал имеет по одному синтезируемому атрибуту *val*, терминал **num** имеет синтезируемый атрибут *ns* (номер строки в таблице символов, предоставляемый лексическим анализатором в качестве атрибута токена числовой константы **num**). Функция *GetVal* (**num**.*ns*) возвращает значение числовой константы из соответствующей строки таблицы символов. Выполняемые правила для продукций очевидны и не требуют дополнительных пояснений. Результатом вычисления значения арифметического выражения является значение атрибута *S.val*, устанавливаемое в правиле *S.val* := *E.val* для первой продукции. Это правило можно заменить правилом с побочным действием, например, вида *Print*(*E.val*) для печати результатов вычислений.

Наглядное представление о процессе вычисления атрибутов дает дерево разбора, в котором в каждом узле показаны значения атрибутов. Такое дерево разбора называется *аннотированным*, а сам процесс вычисления значений атрибутов в узлах дерева разбора – *аннотированием* дерева разбора.

Аннотированное дерево разбора входной строки 1 + 2 * 3, построенное с применением правил из СУО (см. табл. 1), представлено на рис. 1.

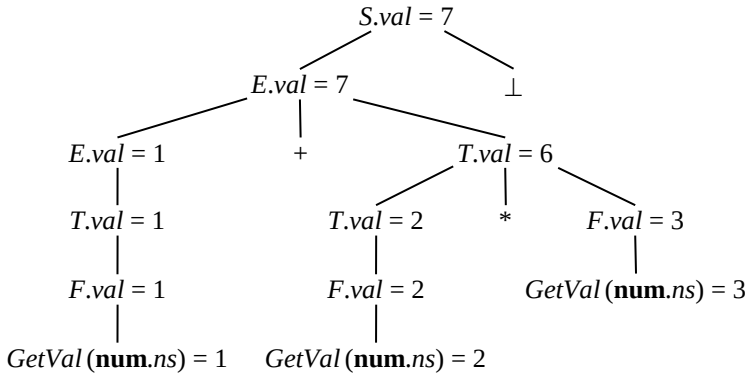

Рис. 1. Аннотированное дерево разбора выражения 1 + 2 * 3

Аннотирование осуществляется выполнением семантических правил в соответствии с обратным прохождением дерева разбора и легко может быть реализовано в процессе восходящего синтаксического анализа, основанного на *LR*-грамматике.

S-атрибутное СУО вычисления арифметического выражения

Продукция	Семантические правила
1) $S \to E\perp$	$S.val := E.val$
2) $E \to E_1 + T$	$E.val := E_1.val + T.val$
3) $E \to T$	$E.val := T.val$
4) $T \to T_1 * F$	$T.val := T_1.val * F.val$
5) $T \to F$	$T.val := F.val$
6) $F \to (E)$	$F.val := E.val$
7) $F \to$ **num**	$F.val :=$ *GetVal* (**num**.*ns*)

Наследуемые атрибуты могут быть полезными, если возникает необходимость распространения информации от предков к потомкам. Несмотря на то, что всегда можно преобразовать СУО так, чтобы в нем использовались только синтезируемые атрибуты, тем не менее, более естественным часто является использование наследуемых атрибутов. Пример СУО с наследуемым атрибутом представлен в табл. 2, а аннотированное дерево разбора строки **char id$_1$, id$_2$** – на рис. 2.

Таблица 2

СУО с наследуемым атрибутом *L.inh*

Продукция	Семантические правила
1) $D \to T\ L$	*L.inh* := *T.type*
2) $T \to$ **int**	*T.type* := integer
3) $T \to$ **char**	*T.type* := char
4) $L \to L_1$ **, id**	L_1*.inh* := *L.inh* *AddType* (**id**.*ns*, *L.inh*)
5) $L \to$ **id**	*AddType* (**id**.*ns*, *L.inh*)

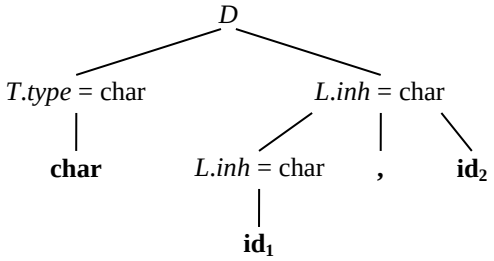

Рис. 2. Аннотированное дерево разбора строки **char id$_1$, id$_2$**

В этом СУО нетерминал *T* имеет синтезируемый атрибут *type*, значением которого является тип объявляемых данных, нетерминал *L* имеет наследуемый атрибут *inh* (от inherit – наследовать), процедура *AddType* записывает тип идентификатора в таблицу символов. После чтения ключевого слова **char** по правилу *T.type* := char, связанному с продукцией 3, устанавливается значение атрибута *T.type* равным char. В правиле *L.inh* := *T.type*, связанному с продукцией 1, наследуемому атрибуту *L.inh* устанавливается значение char, что говорит о том, что все идентификаторы списка *L* будут иметь данный тип. Затем в нисходящем порядке это значение передается в два узла с нетерминалом *L* правого поддерева. В каждом таком узле выполняется также процедура *AddType*, которая записывает тип соответствующего идентификатора в таблицу символов.