Тема 3. Нисходящий синтаксический анализ

Использование СУТ для реализации *L*-атрибутного СУО в процессе нисходящего синтаксического анализа требует, чтобы лежащая в основе СУО грамматика принадлежала классу *LL*. Преобразование *L*-атрибутного СУО в СУТ для его реализации в процессе *LL*-разбора выполняется в соответствии с правилами, из подразд. 2.1, согласно которым действия по вычислению наследуемых атрибутов нетерминала вставляются перед этим нетерминалом, а действия по вычислению синтезируемых атрибутов размещаются в конце продукции.

Если грамматика не относится к классу *LL*, ее следует преобразовать. Наличие атрибутов вносит некоторые особенности в методы устранения левой рекурсии. Если СУТ содержит только действия, не связанные с вычислением атрибутов, то применим стандартный алгоритм устранения левой рекурсии [1; 5], при использовании которого действия рассматриваются так, как если бы это были терминалы. Если же действия вычисляют значения атрибутов, устранение левой рекурсии возможно, если СУТ является постфиксной.

Рассмотрим общую схему для случая одной рекурсивной продукции и одной нерекурсивной продукции [1]:

$A \to A_1\ B\ \{A.a := g\,(A_1.a,\ B.b)\}$

$A \to C\ \{A.a := f\,(C.c)\}$.

Согласно общему правилу преобразования (без учета атрибутов) должны получить грамматику

$A \to C\ X$

$X \to B\ X \mid \varepsilon$.

С учетом действий, связанных с вычислением атрибутов, получается следующая СУТ:

$A \to C\ \{X.i := f\,(C.c)\}\ X\ \{A.a := X.s\}$

$X \to B\ \{\ X_1.i := g\,(X.i,\ B.b)\}\ X_1\ \{X.s := X_1.s\}$

$X \to \varepsilon\ \{X.s := X.i\}$.

Добавленный нетерминал X имеет наследуемый атрибут $X.i$ и синтезируемый атрибут $X.s$.

Для иллюстрации устраним левую рекурсию в грамматике постфиксной СУТ:

$S \to E\perp \ \{Print\,(E.val)\}$

$E \to E_1 + T \ \{E.val := E_1.val + T.val\}$

$E \to T \ \{E.val := T.val\}$

$T \to (E) \ \{T.val := E.val\}$

$T \to \mathbf{num} \ \{T.val := GetVal\,(\mathbf{num}.pnt)\}$.

В соответствии с рассмотренным выше правилом получится следующая СУТ:

$S \to E\perp \ \{Print\,(E.val)\}$

$E \to T \ \{X.i := T.val\} \ X \ \{E.val := X.s\}$

$X \to + \ T \ \{X_1.i := X.i + T.val\} \ X_1 \ \{X.s := X_1.s\}$

$X \to \varepsilon \ \{X.s := X.i\}$

$T \to (E) \ \{T.val := E.val\}$

$T \to \mathbf{num} \ \{T.val := GetVal\,(\mathbf{num}.pnt)\}$.

Детализируем до операций со стеком атрибутов и окончательно получим: (желтым цветом выделена СУТ с предыдущей страницы)

$S \to E\perp \ \{Print\,(E.val)\}$
$E \to T\ \{X.i := T.val\}\ X\ \{E.val := X.s\}$
$X \to +\ T\ \{X_1.i := X.i + T.val\}\ X_1\ \{X.s := X_1.s\}$
$X \to \varepsilon\ \{X.s := X.i\}$
$T \to (E)\ \{T.val := E.val\}$
$T \to \mathbf{num}\ \{T.val := GetVal\,(\mathbf{num}.pnt)\}.$

$S \to E\perp \ \{Print\,(Pop\,(St))\}$
$E \to T\ X$
$X \to +\ T\ \{t_2 := Pop\,(St);\ t_1 := Pop\,(St);\ Push\,(t_1 + t_2, St)\}\ X_1$
$X \to \varepsilon$
$T \to (E)$
$T \to \mathbf{num}\ \{Push\,(GetVal\,(\mathbf{num}.pnt), St)\}.$

Тот же результат будет получен, если для постфиксной СУТ с детализацией операций со стеком из подразд. 3.1

$S \to E\perp$ $\{Print\,(Pop\,(St))\}$

$E \to E_1 + T$ $\{t_2 := Pop\,(St);\ t_1 := Pop\,(St);\ Push\,(t_1 + t_2,\ St)\}$

$E \to T$

$T \to (E)$

$T \to$ **num** $\{Push\,(GetVal\,(\textbf{num.}pnt),\ St)\}$

применить стандартный алгоритм устранения левой рекурсии (включая и левую факторизацию), когда действия рассматриваются как терминалы.

Напомним этот алгоритм.

Пусть множество продукций содержит леворекурсивные продукции $A \to A\alpha_1 | A\alpha_2 | \ldots | A\alpha_m$ и все остальные A-продукции $A \to \beta_1 | \beta_2 | \ldots | \beta_n$, не являющиеся леворекурсивными. Результатом устранения левой рекурсии с последующей левой факторизацией будут продукции:

$A \to \beta_1 X | \beta_2 X | \ldots | \beta_n X$

$X \to \varepsilon | \alpha_1 X | \alpha_2 X | \ldots | \alpha_m X$

Стек *LL*-анализатора наряду с элементами, представляющими терминалы и нетерминалы, должен хранить также элементы действий. Элемент действий представляет собой указатель на выполняемый код, который инициирует запуск выполнения семантического действия в момент, когда элемент действия окажется в вершине стека в процессе *LL*-разбора.

Если для хранения атрибутов используется стек синтаксического анализатора, то синтезируемые атрибуты нетерминала размещаются в стеке под элементом, представляющим этот нетерминал, а наследуемые атрибуты нетерминала хранятся в стеке вместе с этим нетерминалом. На практике можно использовать специальный стек (стеки) для хранения значений атрибутов, как это рассмотрено в подразд. 2.2. Более подробное изложение вопросов хранения атрибутов с иллюстрациями на примерах можно найти в работе А. Ахо и др. [1].