**Еще несколько определений**

В предыдущих примерах мы уже пользовались такими терминами, как «вывод», «язык, порождаемый грамматикой». Теперь дадим формальные определения для этих и ряда других понятий.

Пусть имеются грамматика G = (Т, N, P, S) и цепочка α1 составленная из терминалов и нетерминалов грамматики G, причем α1 представима в виде

**α1 = γ1αγ2**

где **γ1,γ2**— цепочки терминалов и нетерминалов грамматики G, **α** -непустая цепочка терминалов и нетерминалов грамматики G: **γ1,γ2**∈ (T∪N)\*; **α** ∈ (T∪N)+.

Пусть также среди множества правил Р грамматики G есть правило **α→β**. Тогда подцепочка **α** цепочки **α1** может быть заменена цепочкой **β**, в результате чего будет получена цепочка **α2 = γ1βγ2**

В этом случае говорят, что цепочка **α2** непосредственно выводится (порождается) из цепочки **α1** в грамматике G, что записывается следующим образом:

**α1** $⇒$ **α2**, или просто **α1** $⇒$ **α2**,

если используемая грамматика очевидна.

Если имеется последовательность цепочек **α1**, **α2**,..., **αn** (n > 1), таких что

**α1** $⇒$ **α2**, … $⇒$ **αn**, (\*)

то говорят, что цепочка **αn** ,нетривиально выводится из **α1** в грамматике G (выводится за один или более шагов), что обозначается так:

**α1** $⇒$ **αn** или просто **α1** $⇒$ **αn**.

 **G**

Последовательность цепочек **α1**, **α2**,...,**αn** в этом случае называется выводом цепочки **αn** из **α1** в грамматике G.

В дальнейшем выводом будем называть также запись, подобную (\*).

Используется также запись

**α1** $⇒$ **αn** или **α1** $⇒$ **αn**.

 **G**

означающая, что цепочка **αn** выводится из **α1** в грамматике G (выводится за ноль или более шагов), что следует понимать так, что либо **αn** совпадает с **α1**, либо

**α1** $⇒$ **αn**

***Сентенциальной формой*** грамматики G называется цепочка, выводимая из начального нетерминала грамматики G.

Цепочка **α** есть сентенциальная форма грамматики G, если S $⇒$ **α**.

 **G**

***Сентенцией*** (от sentence —предложение) грамматики G называется сентенциальная форма, состоящая только из терминальных символов.

***Язык, порождаемый грамматикой***, есть множество всех ее сентенций.

Можно сказать, что сентенции грамматики — это предложения порождаемого ею языка.

**Дерево вывода**

В последующем рассмотрении предполагается, что мы имеем дело с

грамматиками, все правила которых в своей левой части содержат единственный нетерминал.

Именно такие грамматики, называемые контекстно-свободными, представляют для нас наибольший практический интерес.

Рассмотрим грамматику

G6: S→AB (1)

 А→аА|а (2)

 B→bB|b (3)

Выполним вывод цепочек в этой грамматике. Вначале используем правило (1):

S⇒AB

Будем сопровождать процесс вывода построением дерева (рис. 1).

Корнем дерева будет вершина, соответствующая начальному нетерминалу S. Дочерними

вершинами корня будут вершины А и B, соответствующие правой части первого

примененного правила (рис.1,а). Вершины А и В следуют в дереве слева направо в том же порядке, что и в правиле (1): слева — A, справа — В.



Рис.1 Построение дерева вывода.

Продолжая вывод, мы можем выбрать как правило для нетерминала А — правило (2), так и правило для В — правило (3). Используем вначале первую часть правила (2) (А → аА):

S⇒AB⇒aAB

и продолжим построение дерева (рис. 1, б). Теперь выполним подстановку вместо нетерминала В цепочки bВ по правилу (3):

S ⇒ АВ ⇒ аАВ ⇒ аАbВ ,

добавив к имеющейся вершине В дочерние вершины b и В (рис. 1, в). Еще раз применим правило А → аА и, чтобы получить цепочку, состоящую только из терминалов, выполним подстановки по правилам A →а и В → b. После этого вывод приобретет следующий вид:

S⇒АВ⇒аАВ⇒аАbВ⇒ааАbВ⇒aaabB⇒ aaabb,

а получившееся дерево показано на рис. 1, г.

Получившееся дерево называется деревом вывода, деревом разбора или синтаксическим деревом. Его корень — начальный символ грамматики, внутренние вершины — нетерминалы, листья дерева (концевые вершины) — терминалы.

Обход листьев дерева слева направо дает цепочку терминалов, выведенную из начального символа грамматики (сентенцию).

Нетрудно заметить, что построенное нами дерево будет соответствовать и другим выводам цепочки aaabb в грамматике G6. Вот один из них:

S⇒АВ⇒АbВ⇒аАbВ⇒aAbb⇒aaAbb⇒ aaabb.

Рассмотрение дерева вместо вывода позволяет игнорировать порядок применения правил, если он не важен.

**Задача разбора**

Задача разбора состоит в восстановлении дерева вывода для заданной сентенции.

Разбор — это построение вывода для заранее заданной цепочки. Другими словами, разбор — это тот же вывод, прослеженный в обратном порядке.

Последовательность сентенциальных форм, приводящая к цепочке терминалов (сентенции, предложению языка, порождаемого грамматикой), определяет структуру этой цепочки. Дерево вывода представляет структуру цепочки нагляднее независимо от последовательности применения правил.

Результатом решения задачи разбора в случае, если удалось восстановить дерево для заданной терминальной цепочки, является выявление структуры этой цепочки.

**Построенное дерево называется деревом разбора**.

Успешное восстановление дерева разбора для заданной цепочки означает, что эта цепочка есть правильное предложение языка, порождаемого грамматикой.

Наоборот, если для некоторой цепочки терминалов дерево разбора в данной грамматике построить невозможно, это значит, что цепочка не принадлежит порождаемому грамматикой языку.

**Для чего надо решать задачу разбора**

Разбор (по-английски — parsing) называют также распознаванием или синтаксическим анализом.

Синтаксический анализ имеет две цели — выяснение принадлежности цепочки языку и выявление ее структуры.

Работа любого транслятора основана на распознавании структуры предложений транслируемого языка.

***Синтаксический анализ*** — обязательная фаза в работе компиляторов и интерпретаторов языков программирования.

***Синтаксический анализатор*** — это часть транслятора, составляющая его основу.

Только распознавая структуру входной программы, определяя наличие или отсутствие отдельных ее частей — описаний, операторов, выражений и подвыражений — транслятор может выполнить работу по переводу программы на другой язык.

Часто трансляторы в явном виде строят дерево программы, которое представляется внутренними динамическими структурами данных транслятора, а затем используется при формировании эквивалентной выходной программы.

Если в ходе распознавания дерево вывода не строится, оно присутствует неявно, отражаясь в последовательности выполняемых синтаксическим анализатором действий.

Рассмотрению способов решения задачи разбора — синтаксического анализа будет посвящена большая часть наших занятий.

**Домино Де Ремера**

Де Ремер (De Remer F. L.) предложил наглядную интерпретацию задачи разбора, представив ее как игру в своеобразное домино.

Играющий располагает «костями» домино нескольких типов. Типов столько, сколько правил в грамматике. Каждое правило дает один тип пластинки. Типы домино для грамматики G6 показаны на рис. 2.

Считается, что «костяшек» каждого типа имеется сколько необходимо.



Рис.2 Домино Де Ромера для грамматики G6

Верхняя часть каждого домино соответствует левой части правила грамматики, нижняя — правой. Верхняя и нижние пластинки соединены «резиновыми» нитями. Пластинки можно приставлять друг к другу плоскими сторонами полукруга, если на них записаны одинаковые символы. Фигуры домино нельзя переворачивать, и нельзя менять порядок следования символов (перекрещивать нити).

В начале игры в верхней части поля помещается полукруг, обращенный выпуклостью вверх, в котором записан начальный нетерминал грамматики.

В нижней части игрового поля в полукругах, обращенных плоской частью вверх, размещаются терминальные символы распознаваемой цепочки.

На рис. 3 показана начальная конфигурация игры для цепочки aaabb.



Рис.3 Начало игры

Цель состоит в том, чтобы соединить с помощью имеющихся фигур символы терминальной цепочки и начальный нетерминал. Полученная конфигурация домино для цепочки aaabb и грамматики G6 (набор домино на рис. 2) показана на рис. 4.



Рис.4 Дерево разбора, построенное из домино Де Ремера