Теория формальных языков и компиляторов

Часть 1. Порождающие грамматики и языки

Лекция 4. Языки порождающих грамматик

Вернемся к рассмотрению примера 3.7:

Пример 3.7. Рассмотрим грамматику G4[Z = < целое без знака >] с правилами:

1) <целое без знака> $\rightarrow$ <цифра>

2) <целое без знака> $\rightarrow$ <цифра> <целое без знака>

3) <цифра> $\rightarrow 0|1|2|3|\ldots|9$

Теперь поставим задачу анализа строк: принадлежит ли цепочка «123» множеству <целых без знака >?

Анализ проведём, используя определение *выводимости*:

<целое без знака> $\Rightarrow^{(2)}$ <цифра> <целое без знака> $\Rightarrow^{(2)}$

$\Rightarrow^{(2)}$ <цифра> <цифра> <целое без знака> $\Rightarrow^{(1)}$

$\Rightarrow^{(1)}$ <цифра> <цифра> <цифра> $\Rightarrow^{(3)}$

$\Rightarrow^{(3)}$ «<u>123</u>»

Используя правила грамматики, мы вывели из начального нетерминального символа <целое без знака> строку из терминальных символов. Следует отметить, что 123 в данном случае рассматривается не как целое число, а как синтаксическая конструкция, которая состоит из терминальных символов – цифр.

Определение 4.1. Языком L(G[Z]) называется множество *терминальных* строк (цепочек) *x*, порождаемых грамматикой G[Z], таких что *x* итерационно выводится из начального символа Z.

Цепочки *x* называются *конечной сентенциальной формой*.

Строгое математическое определение языка имеет вид:

$$L\left(G[Z]\right) = \left\{x, Z \Rightarrow^{*} x, x \in V_T^{*}\right\}.$$

Обратите внимание, что здесь выполняется *итерация* над терминальным словарем. Это приводит к возможности существования пустых цепочек в языке.

4.1 Операции над языками

Согласно определению 4.1, язык представляет собой множество. Определим операции над этим множеством.

Определение 4.2. Объединением языков L_1 и L_2 называется новый язык, который включает строки как языка L_1, так и языка L_2.

$$L_1 \cup L_2 = \{x \mid x \in L_1, x \in L_2\}.$$

Определение 4.3. Конкатенацией языков L_1 и L_2 называется новый язык, строки которого образуются конкатенацией строк из L_1 и L_2.

$$L_1 L_2 = \{xy \mid x \in L_1, y \in L_2\}$$

Длина строк нового языка равна сумме длин строк x и y.

Определение 4.4. Итерация языка L представляет новый язык L*, который включает строки из всевозможных комбинаций строк из L любой длины (включая пустую цепочку).

$$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots \cup L^n \cup \ldots.$$

Определение 4.5. Усеченной итерацией языка L является новый язык L^+, который включает строки из всевозможных комбинаций строк из L любой длины, начиная с 1 (исключая пустую цепочку).

$$L^+ = \bigcup_{i=1}^{\infty} L^i.$$

Содержательная интерпретация нового языка L^+ та же, что и для L*, только без пустой цепочки.

Для произвольного формального языка L справедливо утверждение:

$$L = \{\varepsilon\} L = L \{\varepsilon\}.$$

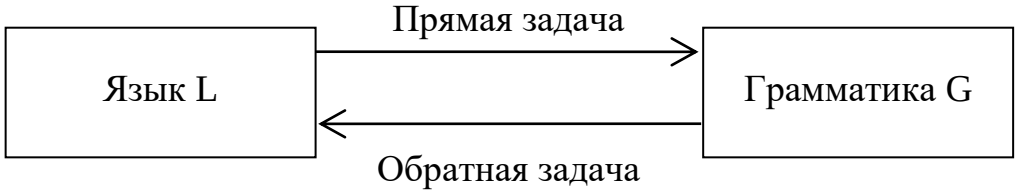
Обратите внимание, что множество, содержащее пустую строку (ε), существенно отличается от пустого множества (Ø). Действительно, в операциях над множествами можно записать:

$$L\varnothing = \varnothing L = \varnothing .$$

Таким образом, новый язык может порождаться не только грамматикой, но и операциями над исходным языком.

4.2 Прямая и обратная задачи формальных языков и грамматик

Задача построения грамматик по языку является прямой задачей. Соответственно выбор языка по грамматике является обратной задачей.



4.2.1 Обратная задача

Рассмотрим примеры обратных задач.

Пример 4.1: Пусть грамматика $G_1[A]$ задана набором:

$V_T = \{0,1\}$,

$V_N = \{A\}$,

P: $A \rightarrow 01$

$A \rightarrow 0A1$

или через альтернативную конструкцию: $A \rightarrow 01 \mid 0A1$.

Требуется определить язык, порождаемый грамматикой $G_1[A]$, т.е. $L(G_1[A])$ – ?

Определим множество цепочек, порождаемых грамматикой $G_1[A]$. Для этого продемонстрируем процедуру выводимости в соответствии с правилами вывода P:

$$\begin{array}{ccccccc} A & \Rightarrow & 0A1 & \Rightarrow & 00A11 & \Rightarrow & 000A111 \quad \ldots \\ \Downarrow & & \Downarrow & & \Downarrow & & \\ 01 & & 0011 & & 000111 & & \ldots \end{array}$$

Очевидно, что это множество включает вполне упорядоченный набор цепочек: { 01,0011,000111,...}. Отсюда легко видеть, что^

$$L(G1[A]) = \{0^n1^n \mid n>=1\}.$$

Определение 4.6. Грамматики G и G_1 являются *эквивалентными*, если они порождают один и тот же язык, т.е. имеет место тождество множеств $L(G) = L(G_1)$.

Определение 4.7. Грамматики G и G_1 являются *почти эквивалентными*, если порождаемые ими языки отличаются не более чем на пустую цепочку ε, т.е. имеет место $L(G) = \{L(G_1) \cup \{\varepsilon\}\}$.

Пример 4.2: Рассмотрим грамматику $G_1[A]$ из примера 4.1 и грамматику $G_2[A]$, которая имеет свои правила P:

$$A \to 0B1$$

$$0B \to 00B1$$

$$B \to \varepsilon.$$

Тем же способом выводимости можно показать эквивалентность грамматик $G_1[A]$ и $G_2[A]$. Действительно,

$$\begin{array}{ccccccc} A & \Rightarrow & 0B1 & \Rightarrow & 00B11 & \Rightarrow & 000B111 \quad \ldots \\ & & \Downarrow & & \Downarrow & & \Downarrow \\ & & 01 & & 0011 & & 000111 \quad \ldots \end{array}$$

Грамматика $G_1[A]$ обладает определенными преимуществами перед $G_2[A]$:

Во-первых, в грамматике $G_1[A]$ имеется меньшее количество нетерминальных символов и меньшее количество продукций. А это связано с меньшим количеством шагов при выводимости определенных цепочек языка.

Во-вторых, отсутствие правил вида $B \to \varepsilon$ делает грамматику более определенной.

Наконец, при разработке грамматик следует стремиться, чтобы в левой части продукций стоял нетерминальный символ, а не смешанная цепочка из терминалов и нетерминалов, как это приведено во втором правиле $G_2[A]$.

Пример 4.3: Пусть $G_3[Z]$ определена на множестве правил вывода P:

$$Z \to aA$$

$$A \to bB$$

$$B \to c$$

A → Bb.

По аналогии с предыдущим примером воспользуемся выводимостью и проследим возможные цепочки языка:

$$Z \Rightarrow aA \Rightarrow abB \Rightarrow abc$$
$$\Downarrow$$
$$aBb \Rightarrow acb$$

Поэтому легко видеть, что язык $L(G_3[Z])$ включает только 2 терминальные цепочки, а именно: $L(G_3[Z]) = \{abc, acb\}$.

Пример 4.4: Определим язык, порождаемый грамматикой $G_4[Z]$:

$V_T = \{a, b\}$,

$V_N = \{A, Z\}$,

P = { Z → aA; A → Ab }.

Вновь воспользуемся выводимостью, так что:

$$Z \Rightarrow aA \Rightarrow aAb \Rightarrow aAbb \Rightarrow \ldots$$

Таким образом, никогда не удастся избавиться от нетерминала A, поэтому грамматика $L(G_4[Z])$ не порождает терминальных цепочек, а язык $L(G_4[Z])$ является пустым $L(G_4[Z]) = Ø$.

Из приведенных примеров следует, что грамматики могут порождать языки, включающие бесконечное множество строк (G_1 и G_2), ограниченное множество цепочек (G_3), и вообще не иметь терминальных цепочек (G_4).

Упражнения

1. Сравните операции над языками и операции со словарями (лекция 2).

2. Определите языки, порождаемые следующими грамматиками:

2.1. P: 1) Z → 11XY0 2) X → 1X | 1 | ε 3) Y → 1Y0 | ε

2.2. P: 1) Z → XY – X 2) X → 1X | 1 | ε 3) Y → 1Y0 | ε

2.3. P: 1) Z → AB 2) A → 1A0 | 10 3) B → 1B0 | 1

2.4. P: 1) Z → A + B 2) A → aA | ε 3) B → bB | ε

2.5. P: 1) Z → A – B 2) A → 1A00 | 10 3) B → aB | ε

2.6. P: 1) $I \to AA$ 2) $A \to a$ 3) $A \to aa$

2.7. P: 1) $I \to aABc$ 2) $I \to \$$ 3) $A \to Ab$

4) $A \to cIB$ 5) $B \to bB$ 6) $B \to a$

2.8. P: 1) $I \to aM$ 2) $M \to A$ 3) $A \to aA$

4) $A \to B$ 5) $B \to bB$ 6) $B \to b$

2.9. P: 1) $I \to aA$ 2) $I \to Ic$ 3) $I \to Ab$ 4) $A \to d$

Список использованных источников

1. Шорников Ю.В. Теория и практика языковых процессоров.