Лекция 2.

Грамматики

§ 2.1. Мотивировка

Первоначально понятие грамматики было формализовано лингвистами при изучении естественных языков. Они интересовались не только определением, что является или не является правильным предложением языка, но также искали способы описания структуры предложений.

Одной из целей была разработка формальной грамматики, способной описывать естественный язык.

Надеялись, что, заложив в компьютер формальную грамматику, например, английского языка, можно сделать его “понимающим” этот язык, осуществлять с помощью компьютера перевод с одного языка на другой, по словесной формулировке проблем получать их решения, и т. д.

По-настоящему хорошего решения этих проблем мы пока не имеем. Но вполне удовлетворительные результаты достигнуты в описании и реализации языков программирования.

Из школьного опыта известно, что собой представляет *грамматический разбор* предложения. При таком разборе определяется, какое слово является подлежащим, какое используется в роли сказуемого, какие слова играют роль определения, дополнения, обстоятельства и т. д.

При разборе мы имеем дело с *грамматическими категориями*: ‘предложение’, ‘группа существительного’, ‘группа сказуемого’, ‘существительное’, ‘глагол’, ‘наречие’ и т. д. и пользуемся собственно *словами*, составляющими разбираемое предложение.

Например, структуру английского предложения: “The little boy ran quickly” можно изобразить в виде диаграммы (рис. 2.1).

Return 86

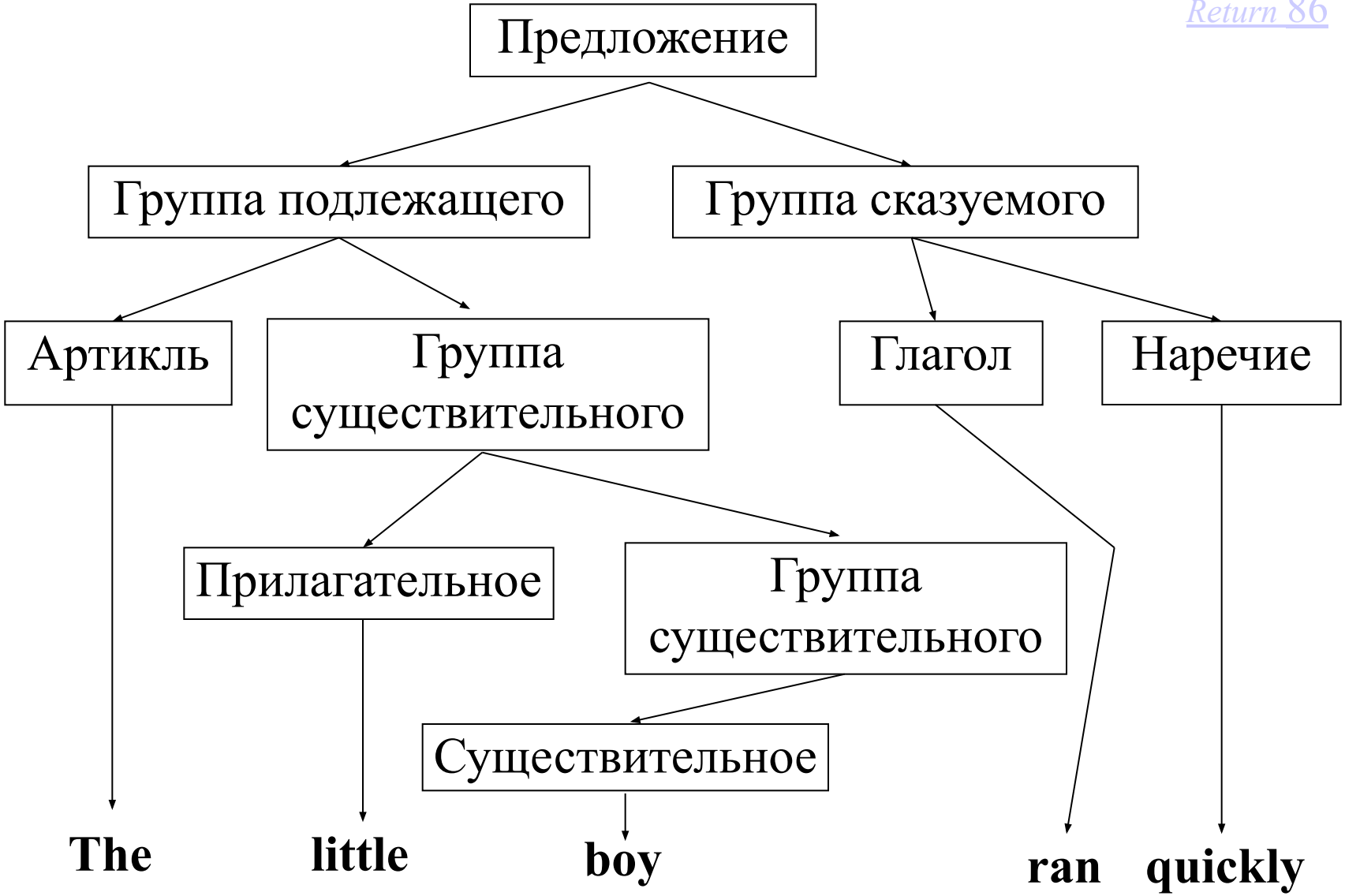

Рис. 2.1. Синтаксическая структура предложения

Грамматический разбор предложений подразумевает использование *правил* некоторой грамматики. Мы их будем представлять в следующей форме (приведены не все правила грамматики):

< предложение > → < группа подлежащего >
< группа сказуемого >

< группа подлежащего > → < артикль >
< группа существительного >

< группа существительного > → < существительное >

< группа сказуемого > → < глагол > < наречие >

< артикль > → The

< прилагательное > → little

< существительное > → boy

< глагол > → ran

< наречие > → quickly

Здесь стрелочка → отделяет *левую часть правила* от *правой*, а грамматические термины заключены в *металингвистические скобки* < и > для того, чтобы отличать их от слов, составляющих разбираемое предложение.

По этим правилам можно не только *проверять грамматическую правильность предложений*, но также *порождать грамматически правильные предложения*.

Механизм порождения

Начиная с цепочки, включающей только грамматический термин, являющимся *главным* (< предложение >), каждый грамматический термин, входящий в текущую цепочку, замещается правой частью того правила, которое содержит его в левой части. Когда в результате таких замен в текущей цепочке не останется ни одного термина грамматики, а только слова языка, мы получаем *грамматически правильное предложение* языка.

§ 2.2. Грамматика. Язык, порождаемый грамматикой

В предыдущем параграфе речь шла о конкретной грамматике. В ней имеются два словаря:

1) *нетерминалы* — грамматические термины

<предложение>, <группа подлежащего >, …;

2) *терминалы* — слова, составляющие предложения языка

The, little, boy, ran, quickly;

3) *правила*, *левые* и *правые* части которых состоят из нетерминалов и терминалов;

<предложение> → < группа подлежащего >
< группа сказуемого >

< артикль > → The, ...

4) *начальный нетерминал* — главный грамматический термин; из него выводятся те цепочки терминалов, которые считаются предложениями языка

<предложение>

Определение 2.1. *Грамматикой* называется четверка $G = (V_N, V_T, P, S)$, где V_N, V_T — *алфавиты* (*словари*) *нетерминалов* и *терминалов* соответственно, причём $V_N \cap V_T = \varnothing$, P — конечное *множество правил*, каждое из которых имеет вид

$$\alpha \to \beta,$$

где $\alpha \in V^* V_N V^*$, $\beta \in V^*$, $V = V_N \cup V_T$ — *объединённый алфавит* (*словарь*) грамматики; S — *начальный нетерминал*.

Определение 2.2. Пусть $\alpha \to \beta \in P$ — правило, а γ, δ — любые цепочки из множества V^*.

Тогда $\gamma\alpha\delta \underset{G}{\Rightarrow} \gamma\beta\delta$ — из цепочки $\gamma\alpha\delta$ *непосредственно выводится* цепочка $\gamma\beta\delta$ в грамматике G при помощи данного правила.

Определение 2.3. Пусть $\alpha_1, \alpha_2, \ldots, \alpha_m$ — цепочки из множества V^* и

$$\alpha_1 \underset{G}{\Rightarrow} \alpha_2, \alpha_2 \underset{G}{\Rightarrow} \alpha_3, \ldots, \alpha_{m-1} \underset{G}{\Rightarrow} \alpha_m.$$

Тогда мы пишем:

$$\alpha_1 \overset{*}{\underset{G}{\Rightarrow}} \alpha_m.$$

и говорим, что “из α_1 *выводится* α_m в грамматике G”.

Другими словами, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, если цепочка β может быть получена из цепочки α путем применения некоторого числа правил из множества правил P.

По определению считается, что $\alpha \overset{*}{\underset{G}{\Rightarrow}} \alpha$ для любой цепочки $\alpha \in V^*$ (*рефлексивность*) и для этого не требуется никаких правил.

Значок $\overset{*}{\underset{G}{\Rightarrow}}$ обозначает *рефлексивно-транзитивное замыкание* отношения непосредственной выводимости $\underset{G}{\Rightarrow}$.

Если мы хотим подчеркнуть, что такой вывод использует по крайней мере одно правило грамматики, то мы пишем: $\alpha \overset{+}{\underset{G}{\Rightarrow}} \beta$.

Значок $\overset{+}{\underset{G}{\Rightarrow}}$ обозначает *транзитивное замыкание* отношения непосредственной выводимости.

Если мы хотим указать, что такой вывод происходит за n шагов, т. е. посредством применения n правил грамматики, то пишем:

$$\alpha \overset{n}{\underset{G}{\Rightarrow}} \beta.$$

Значок $\overset{n}{\underset{G}{\Rightarrow}}$ обозначает n-ю *степень* отношения непосредственной выводимости.

Напомним, что для любого отношения R имеют место следующие тождества:

$$R^0 = E = \{(\alpha, \alpha) \mid \forall\ \alpha \in V^*\},$$

$$R^n = RR^{n-1} = R^{n-1}R \text{ для } n > 0;$$

в частности, $R^1 = R$;

$$R^* = \bigcup_{k=0}^{k=\infty} R^k, \qquad R^+ = \bigcup_{k=1}^{k=\infty} R^k$$

Они, разумеется, применимы и к отношению непосредственной выводимости $\underset{G}{\Rightarrow}$.

Определение 2.4. *Язык, порождаемый грамматикой* G, определим как

$$L(G) = \{w \mid w \in V_{\mathrm{T}}^*,\ S \overset{*}{\underset{G}{\Rightarrow}} w\}.$$

Другими словами, язык есть множество терминальных цепочек, выводимых из начального нетерминала грамматики.

***Определение* 2.5.** Любая цепочка α, такая, что

$$\alpha \in V^* \quad \text{и} \quad S \overset{*}{\underset{G}{\Rightarrow}} \alpha,$$

называется *сентенциальной формой*.

***Определение* 2.6.** Грамматики G_1 и G_2 называются *эквивалентными*, если

$$L(G_1) = L(G_2).$$

Пример 2.1.

Return 23
Return 36

Рассмотрим грамматику $G = (V_N, V_T, P, S)$,
где $V_N = \{S\}$, $V_T = \{0, 1\}$, $P = \{S \to 0S1\ (1),\ S \to 01\ (2)\}$.

Здесь S — единственный *нетерминал*, он же — *начальный*;
0 и 1 — *терминалы*; *правил* два: $S \to 0S1$ и $S \to 01$.

Так как оба правила имеют в левой части по одному символу S, то единственно возможный порядок их применения — сколько-то раз использовать первое правило, а затем один раз использовать второе. Применив первое правило n – 1 раз, а затем второе правило, получим:

$$S \underset{G}{\overset{(1)}{\Rightarrow}} 0S1 \underset{G}{\overset{(1)}{\Rightarrow}} 00S11 \underset{G}{\overset{(1)}{\Rightarrow}} 0^3S1^3 \underset{G}{\overset{(1)}{\Rightarrow}} \ldots \underset{G}{\overset{(1)}{\Rightarrow}} 0^{n-1}S1^{n-1} \underset{G}{\overset{(2)}{\Rightarrow}} 0^n1^n.$$

Здесь мы воспользовались обозначением $w^i = \underbrace{w \ldots w}_{i \text{ раз}}$,
причем $w^0 = \varepsilon$.

Таким образом, эта грамматика порождает язык

$$L(G) = \{\, 0^n1^n \mid n > 0\}.$$

§ 2.3. Типы грамматик

Грамматику, определённую в предыдущем параграфе, вслед за Н. Хомским назовем *грамматикой типа* 0.

Им введено ещё три типа грамматик, различающихся ограничениями, накладываемыми на вид правил.

***Определение* 2.7.** Грамматика

$$G = (V_N, V_T, P, S)$$

является *грамматикой типа* 1 или *контекстно-зависимой грамматикой*, если для каждого её правила $\alpha \to \beta \in P$ выполняется условие $|\beta| \geq |\alpha|$.

Часто вместо термина “контекстно-зависимая грамматика” используют аббревиатуру csg (**c**ontext-**s**ensitive **g**rammar).

Очевидно, что грамматику типа 0, приведенную в примере 2.1, можно считать также и контекстно-зависимой грамматикой, поскольку правые части её правил не короче левых частей.

Пример 2.2. Пусть $G = (V_N, V_T, P, S)$, где $V_N = \{S, B, C\}$, $V_T = \{a, b, c\}$, $P = \{(1)\ S \to aSBC,$ $(2)\ S \to aBC,\ (3)\ CB \to BC,\ (4)\ aB \to ab,$ $(5)\ bB \to bb,\ (6)\ bC \to bc,\ (7)\ cC \to cc\}$.

Язык $L(G)$ содержит цепочки вида $a^n b^n c^n$ для каждого $n \geq 1$, так как

1: мы можем использовать правило (1) $n - 1$ раз.
2: Затем мы используем правило (2).

3: Применим правило (3) $m = n(n–1)/2$ раз. Тогда все B станут предшествовать всем C.

4: Далее мы используем один раз правило (4).

5: Затем, используем правило (5) $n - 1$ раз.

6: Применим один раз правило (6).
7: Наконец, применим $n - 1$ раз правило (7).

$$S \underset{G}{\overset{(1)}{\Rightarrow}}{}^{n-1} a^{n-1}S(BC)^{n-1} \underset{G}{\overset{(2)}{\Rightarrow}} a^n(BC)^n \underset{G}{\overset{(3)}{\Rightarrow}}{}^{m} a^nB^nC^n \underset{G}{\overset{(4)}{\Rightarrow}} a^nbB^{n-1}C^n \underset{G}{\overset{(5)}{\Rightarrow}}{}^{n-1} a^nb^nC^n \underset{G}{\overset{(6)}{\Rightarrow}} a^nb^ncC^{n-1} \underset{G}{\overset{(7)}{\Rightarrow}}{}^{n-1} a^nb^nc^n .$$

Retun 27 Retun 667 RetunRetu n 885

Замечание 2.1. Некоторые авторы требуют, чтобы правила контекстно-зависимой грамматики имели вид:

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2,$$

где $\alpha_1, \alpha_2, \beta \in V^*$, причем $\beta \neq \varepsilon$, а $A \in V_N$.

Это мотивирует название “контекстно-зависимая”, так как правило $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$ позволяет заменять A на β только, если A появляется в сентенциальной форме в контексте α_1 и α_2.

В отечественной литературе для таких грамматик чаще используется термин НС-грамматики — *грамматики непосредственных составляющих*, а грамматики типа 1 называются *неукорачивающими грамматиками*.

Грамматика, приведённая в примере 2.2, не является НС-грамматикой из-за того, что правило $CB \rightarrow BC$ не имеет вида:

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2.$$

Действительно, если считать, что левая часть этого правила имеет вид: εCB, то правая часть может быть устроена лишь по образцу: $\varepsilon \ldots B$. Никакая подстановка вместо многоточия не может дать BC. Если взять за образец левой части $CB\varepsilon$, то правая часть должна иметь вид: $C \ldots \varepsilon$, и опять никакая замена многоточия не даст BC.

Теорема 2.1. *Классы языков, порождаемых неукорачивающими и НС-грамматиками, равны.*

Доказательство. Во-первых, любая НС-грамматика является неукорачивающей.

С другой стороны, любое правило неукорачивающей грамматики может быть преобразовано так, чтобы все символы, его составляющие, были нетерминалами.

Для этого достаточно

- каждое вхождение терминала $a \in V_{\text{T}}$ заменить на новый нетерминал Z,
- пополнить словарь нетерминалов этим символом и
- включить правило $Z \to a$ в множество правил грамматики.

Правила вида $Z \to a$ допустимы для НС-грамматик.

Правило же вида

$$X_1X_2\ldots X_m \to Y_1Y_2\ldots Y_{m+q}$$

неукорачивающей грамматики, где

$m > 0,\ q \geq 0,\ X_i, Y_j \in V_N,\ 1 \leq i \leq m,\ 1 \leq j \leq m+q,$

эквивалентно группе правил:

$X_1X_2X_3\ldots X_m \to A_1X_2X_3\ldots X_m,$

$A_1X_2X_3\ldots X_m \to A_1A_2X_3\ldots X_m,$

…

$A_1A_2A_3\ldots X_m \to A_1A_2A_3\ldots A_m,$

$A_1A_2A_3\ldots A_m \to Y_1A_2A_3\ldots A_m,$

$Y_1A_2A_3\ldots A_m \to Y_1Y_2A_3\ldots A_m,$

…

$Y_1Y_2Y_3\ldots A_{m-1}A_m \to Y_1Y_2Y_3\ldots Y_{m-1}A_m,$

$Y_1Y_2\ldots Y_{m-1}A_m \to Y_1Y_2\ldots Y_{m-1}Y_mY_{m+1}\ldots Y_{m+q},$

где $A_1, A_2, \ldots, A_m$ — дополнительные нетерминалы.

Замечание 2.2. Отметим, что новые правила с нетерминалами A могут применяться только в *одной* указанной последовательности и никакого другого эффекта, кроме того, какое производит заменяемое правило, воспроизвести не могут.

Соблазн “оптимизировать” построение НС-грамматики, не используя дополнительные нетерминалы A, а заменяя по-одному нетерминалы левой части X сразу на нетерминалы правой части Y, за исключением последнего правила, заменяющего последний X на все оставшиеся Y, мог бы привести к не эквивалентной грамматике.

Пример 2.3. Покажем на примере, как построить НС-грамматику, эквиваленную неукорачивающей грамматике примера 2.2:

$G = (V_N, V_T, P, S)$, где $V_N = \{S, B, C\}$, $V_T = \{a, b, c\}$,

$P = \{$ (1) $S \to aSBC$, (2) $S \to aBC$, (3) $CB \to BC$, (4) $aB \to ab$, (5) $bB \to bb$, (6) $bC \to bc$, (7) $cC \to cc\}$.

Все правила, кроме (3), не нуждаются ни в каких преобразованиях, т. к. уже соответствуют требованиям НС-грамматики.

Правило (3) заменим группой правил:

$$(3.1)\ CB \to A_1B,\ (3.2)\ A_1B \to A_1A_2,$$

$$(3.3)\ A_1A_2 \to BA_2,\ (3.4)\ BA_2 \to BC,$$

которые все также соответствуют требованиям НС-грамматики. Эти правила не для чего, кроме как для вывода:

$$CB \Rightarrow A_1B \Rightarrow A_1A_2 \Rightarrow\ BA_2 \Rightarrow BC$$

служить не могут. В них A_1 и A_2 новые нетерминалы.

***Определение* 2.8.** Грамматика

$$G = (V_N, V_T, P, S)$$

является *грамматикой типа* 2 или *контекстно-свободной грамматикой*, если каждое её правило имеет вид

$$A \to \beta \in P,$$

где $A \in V_N$, $\beta \in V^+$.

Вместо термина “контекстно-свободная грамматика” часто используют аббревиатуру cfg (**c**ontext-**f**ree **g**rammar) или сокращение КС-грамматика.

Замечание 2.3. Правило вида $A \to \beta$ позволяет заменить A на β независимо от контекста, в котором появляется A.

Грамматика, приведенная в примере 2.1, является не только грамматикой типа 0, грамматикой типа 1, но и контекстно-свободной (по Хомскому типа 2).

Пример 2.3. Рассмотрим интересную контекстно-свободную грамматику

$G = (V_N, V_T, P, S)$, где

$V_N = \{S, A, B\}$, $V_T = \{a, b\}$,

$P = \{S \to aB, S \to bA,$

$A \to a, A \to aS, A \to bAA,$

$B \to b, B \to bS, B \to aBB\}.$

Индукцией по длине цепочки покажем, что

$$L(G) = \{x \in \{a, b\}^+ \mid \#_a x = \#_b x\},$$

где $\#_a x$ обозначает число букв a в цепочке x, а $\#_b x$ — число букв b.

Другими словами, язык, порождаемый этой грамматикой, состоит из непустых цепочек, в которых число букв a и b одинаково.

Заметим, что в порождении языка участвуют *все правила P и только они*.

С учётом этого достаточно доказать для $x \in V_{\text{T}}^{+}$,что

1) $S \overset{*}{\Rightarrow} x$ тогда и только тогда, когда $\#_a x = \#_b x$;

2) $A \overset{*}{\Rightarrow} x$ тогда и только тогда, когда $\#_a x = \#_b x + 1$;

3) $B \overset{*}{\Rightarrow} x$ тогда и только тогда, когда $\#_b x = \#_a x + 1$.

База. Очевидно, что все три утверждения выполняются для всех x: $|x| = 1$, поскольку $A \Rightarrow a$, $B \Rightarrow b$ и никакая терминальная цепочка длины 1 не выводима из S.

Кроме того, никакие цепочки единичной длины, отличающиеся от a и b, не выводимы из A и B соответственно.

Индукционная гипотеза. Предположим, что утверждения 1–3 выполняются для всех x:

$$|x| \le k \ (k \ge 1).$$

Индукционный переход. Покажем, что они выполняются для x: $|x| = k + 1$.

Необходимость.

(1) Если $S \overset{k+1}{\Rightarrow} x$, то вывод должен начинаться

(1.1) либо с правила $S \to aB$,

(1.2) либо с правила $S \to bA$.

В случае (1.1) имеем

$$S \Rightarrow aB \overset{k}{\Rightarrow} ax_1 = x,$$

причём $B \overset{k}{\Rightarrow} x_1$.

По индукционному предположению $\#_b x_1 = \#_a x_1 + 1$, так что $\#_a x = \#_b x$.

Во случае (1.2)

$$S \Rightarrow bA \overset{k}{\Rightarrow} bx_1 = x,$$

причём $A \overset{k}{\Rightarrow} x_1$.

По индукционному предположению $\#_a x_1 = \#_b x_1 + 1$, так что $\#_a x = \#_b x$.

(2) Если $A \overset{k+1}{\Rightarrow} x$, то этот вывод может начаться

(2.1) либо с правила $A \to aS$,

(2.2) либо с правила $A \to bAA$.

В случае (2.1) имеем

$$A \Rightarrow aS \overset{k}{\Rightarrow} ax_1 = x,\ S \overset{k}{\Rightarrow} x_1,$$

так что по индукционной гипотезе

$\#_a x_1 = \#_b x_1$ и, следовательно, $\#_a x - \#_b x = 1$.

В случае (2.2) имеем

$$A \overset{k+1}{\Rightarrow} bAA \overset{k_1}{\Rightarrow} bx_1A \overset{k_2}{\Rightarrow} bx_1x_2 = x;\ A \overset{k_1}{\Rightarrow} x_1,\ A \overset{k_2}{\Rightarrow} x_2,$$

причём $k_1 + k_2 = k$, так что по индукционной гипотезе, поскольку

$$k_1 < k \ \text{ и } \ k_2 < k,$$

выполняются соотношения

$$\#_a x_1 - \#_b x_1 = 1,\ \ \#_a x_2 - \#_b x_2 = 1$$

и, следовательно, $\#_a x - \#_b x = 1$.

Необходимость утверждения (3) доказывается аналогично (2).

Достаточность. Пусть $|x| = k + 1$.

(1) Пусть $\#_a x = \#_b x$.

Либо первый символ x есть a, либо он есть b.

Предположим, что $x = ax_1$.

Теперь $|x_1| = k$ и цепочка $\#_b x_1 = \#_a x_1 + 1$.

По индукционной гипотезе $B \overset{*}{\Rightarrow} x_1$.

Но тогда $S \Rightarrow aB \overset{*}{\Rightarrow} ax_1 = x$.

Аналогичное рассуждение достигает цели, если первый символ x есть b.

Достаточность утверждений (2) и (3) доказывается аналогично.

***Определение* 2.9.** Грамматика

$$G = (V_N, V_T, P, S)$$

является *грамматикой типа* 3 или *регулярной грамматикой* (rg (а-джи) — **r**egular **g**rammar), если каждое её правило имеет вид

$$A \to aB \text{ или } A \to a,$$

где $a \in V_T$; $A, B \in V_N$.

Замечание 2.4.

В лекции 3 будет определено абстрактное устройство, называемое *конечным автоматом*, и показано, что языки, порождаемые грамматиками типа 3, являются в точности теми множествами, которые распознаются (допускаются) этими устройствами. Поэтому такой класс грамматик и языков часто называют *конечно-автоматными* или просто *автоматными*.

Пример 2.4. Рассмотрим грамматику

$G = (V_N, V_T, P, S)$, где $V_N = \{S, A, B\}$, $V_T = \{0, 1\}$,
$P = \{(1)\ S \to 0A,\ (2)\ S \to 1B,\ (3)\ S \to 0,$
$(4)\ A \to 0A,\ (5)\ A \to 0S,\ (6)\ A \to 1B,$
$(7)\ B \to 1B,\ (8)\ B \to 1,\ \ (9)\ B \to 0\}.$

Ясно, что G — регулярная грамматика.

Вопрос: Определить, какой язык порождает данная грамматика?

Ответ на вопрос

- Нетерминал *B* порождает символ 0 или 1, которому может предшествовать любое число 1:

$$1^*(0\ ;\ 1)$$

- Нетерминал *A* порождает:

(1) такие же цепочки, что и *B*, но начальная цепочка из единиц не пуста, и кроме того им может предшествовать любое число 0:

$$0^*1^+(0\ ;\ 1)$$

(2) а также цепочки, какие порождает нетерминал *S*, но с символом 0 в начале:

$$0S$$

- Наконец, нетерминал S порождает:

(1) то же, что A, но с обязательным 0 в начале:

$$0^+1^+(0\ ;\ 1)\ ;\ 00S$$

(2) то же, что B, но с обязательной 1 в начале:

$$1^+(0\ ;\ 1)$$

(3) либо просто 0.

Итак, $L(G) = (00)^*(0\ ;\ 0^*1^+(0\ ;1)) =$

$$=\{0^{2n+1}\ |\ n = 0, 1, 2, ...\}\ \cup\ 0^*1^+(0\ ;1).$$

Классы языков

Очевидно, что

- каждая грамматика типа 3 является грамматикой типа 2;
- каждая грамматика типа 2 является грамматикой типа 1;
- каждая грамматика типа 1 является грамматикой типа 0.

Каждому классу грамматик соответствует класс языков. Языку приписывается тип грамматики, которой он порождается.

Например, контекстно-свободные грамматики (cfg) порождают контекстно-свободные языки (cfl), контекстно-зависимые грамматики (csg) порождают контекстно-зависимые языки (csl).

В соответствии с текущей практикой язык типа 3 или регулярный язык часто называют *регулярным множеством* (rs — **r**egular **s**et).

Язык типа 0 называют *рекурсивно перечислимым множеством* (res — **r**ecur-sively **e**numerable **s**et).

Далее будет показано, что языки типа 0 соответствуют языкам, которые интуитивно могут быть перечислимы конечно описываемыми процедурами.

Очевидно, что $L_3 \subseteq L_2 \subseteq L_1 \subseteq L_0$. Впоследствии мы убедимся, что вложение этих классов языков *строгое*.

§ 2.4. Пустое предложение

Грамматики определены так, что пустое предложение (ε) не находится ни в контекстно-свободном (cfl), ни в контекстно-зависимом (csl) языках, ни в регулярном множестве (rs). Теперь мы расширим данные ранее определения csg, cfg и rg, допустив порождение пустого предложения посредством правила вида $S \rightarrow \varepsilon$, где S — начальный символ, при условии, что S не появляется в правой части никакого правила. В этом случае ясно, что правило $S \rightarrow \varepsilon$ может использоваться только в качестве первого и единственного шага вывода.

Имеет место следующая лемма.

Лемма 2.1. *Если* $G = (V_N, V_T, P, S)$ *есть контекстно-зависимая, контекстно-свободная или регулярная грамматика, то существует другая грамматика* G_1 *такого же типа, которая порождает тот же самый язык и в которой ни одно правило не содержит начальный символ в своей правой части.*

Return 6 62

Доказательство. Пусть S_1 — символ, не принадлежащий ни алфавиту нетерминалов, ни алфавиту терминалов.

Положим $G_1 = (V_N \cup \{S_1\}, V_T, P_1, S_1)$, где $P_1 = P \cup \{S_1 \to \alpha \mid S \to \alpha \in P\}$. Поскольку символ $S_1 \notin V_N$, то он не появляется в правой части никакого правила из множества P_1.

Докажем, что $L(G) = L(G_1)$.

I. Покажем сначала, что $L(G) \subseteq L(G_1)$.

Пусть $x \in L(G)$, т. е. $S \overset{*}{\underset{G}{\Rightarrow}} x$, причем первое используемое правило есть $S \to \alpha \in P$.

Тогда $S \underset{G}{\Rightarrow} \alpha \overset{*}{\underset{G}{\Rightarrow}} x$.

По построению P_1 правило $S_1 \to \alpha \in P_1$, так что $S_1 \underset{G_1}{\Rightarrow} \alpha$.

Поскольку любое правило грамматики G является также правилом грамматики G_1, то $\alpha \overset{*}{\underset{G_1}{\Rightarrow}} x$. Таким образом, имеем $S_1 \underset{G_1}{\Rightarrow} \alpha \overset{*}{\underset{G_1}{\Rightarrow}} x$, то есть $x \in L(G_1)$ и тем самым доказано, что

$$L(G) \subseteq L(G_1).$$

II. Покажем теперь, что $L(G_1) \subseteq L(G)$.

Пусть $x \in L(G_1)$, т. е. $S_1 \underset{G_1}{\overset{*}{\Rightarrow}} x$. Пусть первое используемое правило есть $S_1 \to \alpha \in P_1$, т. е. имеем $S \underset{G_1}{\Rightarrow} \alpha \underset{G_1}{\overset{*}{\Rightarrow}} x$.

Но такое правило существует во множестве P_1 только потому, что в правилах P имеется правило $S \to \alpha$. Следовательно, $S \underset{G}{\Rightarrow} \alpha$.

С другой стороны, $\alpha \overset{*}{\underset{G_1}{\Rightarrow}} x$ и α не содержит символа S_1. Поскольку ни одно правило из множества P_1 не содержит справа символа S_1, то ни одна сентенциальная форма этого вывода также не содержит символа S_1. Значит, в этом выводе используются только такие правила, которые имеются в множестве P. Поэтому $\alpha \overset{*}{\underset{G}{\Rightarrow}} x$.

С учетом того, что $S \underset{G}{\Rightarrow} \alpha$, получаем вывод $S \underset{G}{\Rightarrow} \alpha \overset{*}{\underset{G}{\Rightarrow}} x$. Это и означает, что

$$L(G_1) \subseteq L(G).$$

Из утверждений I и II следует, что

$$L(G) = L(G_1).$$

Очевидно, что грамматики G и G_1 имеют один и тот же тип. Действительно, все правила грамматики G являются правилами грамматики G_1. Те же новые правила, которые имеются в грамматике G_1, но отсутствуют в грамматике G, отличаются от прототипа лишь нетерминалом слева, что не может изменить тип грамматики.

Что и требовалось доказать.

Теорема 2.2. *Если L — контекстно-зависимый, контекстно-свободный или регулярный язык, то языки $L \cup \{\varepsilon\}$, $L \setminus \{\varepsilon\}$ также являются соответственно контекстно-зависимым, контекстно-свободным или регулярным языком.*

Доказательство. Согласно лемме 2.1 существует грамматика G, порождающая язык L, начальный нетерминал которой не встречается в правых частях её правил.

Если язык $L = L(G)$ не содержит пустого предложения, то мы можем пополнить грамматику G ещё одним правилом вида $S \to \varepsilon$.

Обозначим пополненную грамматику G_1.

Правило $S \rightarrow \varepsilon$ может использоваться только как первое и единственное правило вывода в G_1, поскольку начальный нетерминал S не встречается в правых частях правил.

Любой вывод в грамматике G_1, не использующий правило $S \rightarrow \varepsilon$, есть также вывод в G, так что $L(G_1) = L(G) \cup \{\varepsilon\}$.

Если же $L = L(G)$ содержит пустое предложение, то среди правил грамматики G имеется правило вида $S \to \varepsilon$, с помощью которого только пустое предложение и выводится. Ни в каком другом выводе это правило не используется, так что, если его отбросить, то получим грамматику G_1, которая порождает все предложения языка L, кроме пустого. Следовательно,

$$L(G_1) = L(G) \setminus \{\varepsilon\}.$$

Согласно лемме 2.1 типы грамматик G и G_1 одинаковы, поэтому одинаковы и типы языков, порождаемых этими грамматиками.

Что и требовалось доказать.

Пример 2.5.

Перестроим грамматику G из (сл.24) примера 2.2 согласно лемме 2.1 так, чтобы начальный нетерминал не встречался в правых частях правил.

Обозначим перестроенную грамматику G_1:

$G_1 = (V_N, V_T, P_1, S_1)$, где $V_N = \{S_1, S, B, C\}$,

$V_T = \{a, b, c\}$, $P_1 = P \cup \{S_1 \to aSBC, S_1 \to aBC\} =$

$\{(1)\ S \to aSBC, (2)\ S \to aBC, (3)\ CB \to BC,$

$(4)\ aB \to ab, (5)\ bB \to bb, (6)\ bC \to bc,$

$(7)\ cC \to cc, (8)\ S_1 \to aSBC, (9)\ S_1 \to aBC\ \}$

(1–7 — старые правила, 8 – 9 — дополнительные правила).

Пример 2.5.

Построенная грамматика G_1 отличается от исходной грамматики G только дополнительным нетерминалом S_1, используемым в качестве нового начального символа, и двумя дополнительными правилами, его определяющими.

Согласно лемме 2.1

$$L(G_1) = L(G) = \{a^n b^n c^n \mid n \geq 1\}.$$

Мы можем добавить пустое предложение к $L(G_1)$, пополнив грамматику G_1 ещё одним правилом: $S_1 \to \varepsilon$.

Обозначим пополненную грамматику G_2.

Тогда $G_2 = (V_N, V_T, P_2, S_1)$, где

$V_N = \{S_1, S, B, C\}$, $V_T = \{a, b, c\}$,

$P_2 = P_1 \cup \{ S_1 \to \varepsilon \}$.

Очевидно, что

$$L(G_2) = L(G_1) \cup \{\varepsilon\} = \{a^n b^n c^n \,|\, n \geq 0\}.$$

§ 2.5. Рекурсивность контекстно-зависимых грамматик

***Определение* 2.10.** Грамматика

$$G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, S)$$

— *рекурсивна*, если существует алгоритм, который определяет (за конечное время), порождается ли любая данная цепочка $x \in V_{\mathrm{T}}^{*}$ данной грамматикой G.

Пусть $G = (V_N, V_T, P, S)$ — контекстно-зависимая грамматика.

Проверить, порождается пустое предложение данной грамматикой или нет, просто. Достаточно посмотреть, имеется ли в ней правило $S \to \varepsilon$, поскольку $\varepsilon \in L(G)$ тогда и только тогда, когда $S \to \varepsilon \in P$.

Если $\varepsilon \in L(G)$, то мы можем образовать новую грамматику:

$$G_1 = (V_N, V_T, P_1, S),$$

отличающуюся от исходной лишь тем, что в ней не будет правила $S \to \varepsilon$, т. е.

$$P_1 = P \setminus \{S \to \varepsilon\}.$$

Согласно теореме 2.1 грамматика G_1 тоже контекстно-зависима, и $L(G_1) = L(G) \setminus \{\varepsilon\}$.

Следовательно, в любом выводе длина последовательных сентенциальных форм разве лишь возрастает (т. е. не убывает).

Пусть объединенный словарь $V = V_{\mathrm{N}} \cup V_{\mathrm{T}}$ грамматики G_1 имеет k символов.

Предположим, что $S \overset{*}{\underset{G_1}{\Rightarrow}} x$, $x \neq \varepsilon$. Пусть этот вывод имеет вид:

$$S \underset{G_1}{\Rightarrow} \alpha_1 \underset{G_1}{\Rightarrow} \alpha_2 \underset{G_1}{\Rightarrow} \ldots \underset{G_1}{\Rightarrow} \alpha_m = x,$$

причём $|\alpha_1| \leq |\alpha_2| \leq \ldots \leq |\alpha_m|$, и предположим, что

$$|\alpha_i| = |\alpha_{i+1}| = \ldots = |\alpha_{i+j}| = p.$$

Предположим также, что $j \geq k \times p$

Тогда среди этих сентенциальных форм, по крайней мере, какие-то две одинаковы, так как число всевозможных различных непустых цепочек длиной p, составленных из символов алфавита V, в котором k символов, равно $k \times p$. В рассматриваемой же последовательности мы имеем $j + 1$ цепочек, где $j \geq k \times p$.

Пусть, например, $\alpha_q = \alpha_r$, где $q < r$. Тогда

$$S \underset{G_1}{\Rightarrow} \alpha_1 \underset{G_1}{\Rightarrow} \alpha_2 \underset{G_1}{\Rightarrow} \ldots \underset{G_1}{\Rightarrow} \alpha_q \underset{G_1}{\Rightarrow} \alpha_{r+1} \underset{G_1}{\Rightarrow} \ldots \underset{G_1}{\Rightarrow} \alpha_m = x$$

является более коротким выводом цепочки x в грамматике G_1, чем первоначальный.

Интуитивно ясно, что если существует какой-нибудь вывод терминальной цепочки, то существует и “не слишком длинный” её вывод. В следующей теореме описывается алгоритм распознавания, в котором существенно используется это соображение.

Теорема 2.3. *Если грамматика*

$$G = (V_N, V_T, P, S)$$

— контекстно-зависима, то она рекурсивна.

Доказательство. Ранее было показано, что существует простой способ узнать, действительно ли $\varepsilon \in L(G)$, и если это так, то, исключив из грамматики правило $S \rightarrow \varepsilon$, получим грамматику, которая порождает тот же самый язык, но без пустого предложения.

Любое же непустое предложение языка выводимо без использования этого правила. Поэтому, предполагая, что P не содержит правила $S \to \varepsilon$, рассмотрим произвольную цепочку $x \in V_{\mathrm{T}}^{+}$.

Наша задача найти алгоритм, разрешающий вопрос: $x \in L(G)$?

Пусть $|x| = n$ ($n > 0$). Определим множество T_m ($m \geq 0$) следующим образом:

$$T_m = \{\alpha \mid S \overset{i}{\Rightarrow} \alpha,\ i \leq m,\ \alpha \in V^+,\ |\alpha| \leq n\}.$$

Другими словами, T_m содержит сентенциальные формы, выводимые не более, чем за m шагов, и не длиннее, чем n символов.

Очевидно, что

$T_0 = \{S\}$, и при $m > 0$

$$T_m = T_{m-1} \cup \{\alpha \mid \beta \Rightarrow \alpha, \beta \in T_{m-1}, \mid \alpha \mid \leq n\},$$

т. е. T_m есть результат пополнения множества T_{m-1} цепочками, выводимыми из его цепочек за один шаг, длина которых не превосходит n.

Если $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$ и $|\alpha| \le n$, то $\alpha \in T_m$ при некотором m.

Если $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$ не имеет места или $|\alpha| > n$, то $\alpha \notin T_m$ ни при каком m.

Также очевидно, что $T_{m-1} \subseteq T_m$ для всех $m \ge 1$.

Если на некотором шаге вычислений членов последовательности $T_0, T_1, T_2, \ldots$ окажется $T_m = T_{m-1}$, $(m \ge 1)$, и поскольку T_m зависит только от T_{m-1}, то $T_m = T_{m+1} = T_{m+2} = \ldots$.

Наш алгоритм будет вычислять T_1, T_2, T_3, ... до тех пор, пока для некоторого m не окажется $T_m = T_{m-1}$.

Если $x \notin T_m$, то $x \notin L(G)$, потому что $T_j = T_m$ для всех $j > m$.

Конечно, если $x \in T_m$, то $S \overset{*}{\underset{G}{\Rightarrow}} x$.

Осталось доказать, что для некоторого m непременно будет $T_m = T_{m-1}$.

Вспомним, что $T_i \supseteq T_{i-1}$ для каждого $i \geq 1$. При $T_i \neq T_{i-1}$ число элементов во множестве T_i, по крайней мере, на 1 больше, чем во множестве T_{i-1}.

Если алфавит V имеет k символов, то число строк длиной n или меньше в множестве V^+ равно

$k + k^2 + k^3 + ... + k^n <$

$k(k+1)^0 + k(k+1)^1 + k(k+1)^2 + ... + k(k+1)^{n-1} <$

$<(k+1)^n$.

И это единственно возможные строки, которые могут быть в любом множестве T_i.

Таким образом, при некотором

$$m \le (k + 1)^n$$

непременно случится, что $T_m = T_{m-1}$.

Следовательно, процесс вычисления множеств T_i $(i > 0)$ гарантированно закончится за конечное число шагов, и он тем самым является алгоритмом.

Замечание 2.5. Нет нужды доказывать, что алгоритм, описанный в теореме 2.3, применим также к контекстно-свободным и регулярным грамматикам.

Пример 2.6. Рассмотрим грамматику G из примера 2.2.

С помощью только что описанного алгоритма проверим: $abac \in L(G)$?

$|\, abac \,| = 4$.

$T_0 = \{S\}$.

$T_1 = \{S, aSBC, aBC\}$.

$T_2 = \{S, aSBC, aBC, abC\}$.

$T_3 = \{S, aSBC, aBC, abC, abc\}$.

$T_4 = T_3$.

Поскольку $abac \notin T_3$, то $abac \notin L(G)$.

§ 2.6. Деревья вывода в контекстно-свободных грамматиках

Рассмотрим теперь наглядный метод описания любого вывода в контекстно-свободной грамматике. Фактически мы его уже могли наблюдать в §2.1.

***Определение* 2.11.** Пусть

$$G = (V_N, V_T, P, S) \text{ — cfg.}$$

Дерево есть *дерево вывода* в грамматике G, если оно удовлетворяет следующим четырем условиям:

1) каждый узел имеет метку — символ из алфавита V;

2) метка корня — S;

3) если узел имеет по крайней мере одного потомка, то его метка должна быть нетерминалом;

4) если узлы $n_1, n_2, \ldots, n_k$ — прямые потомки узла n, перечисленные слева направо, с метками $A_1, A_2, \ldots, A_k$ соответственно, а метка узла n есть A, то $A \to A_1 A_2 \ldots A_k \in P$.

Пример 2.7. Рассмотрим КС-грамматику

$$G = (\{S, A\}, \{a, b\}, P, S),$$

где

$$P = \{(1)\ S \to aAS,\ (2)\ S \to a,$$
$$(3)\ A \to SbA,\ (4)\ A \to ba,\ (5)\ A \to SS\}.$$

Деревья вывода в контекстно-свободных грамматиках

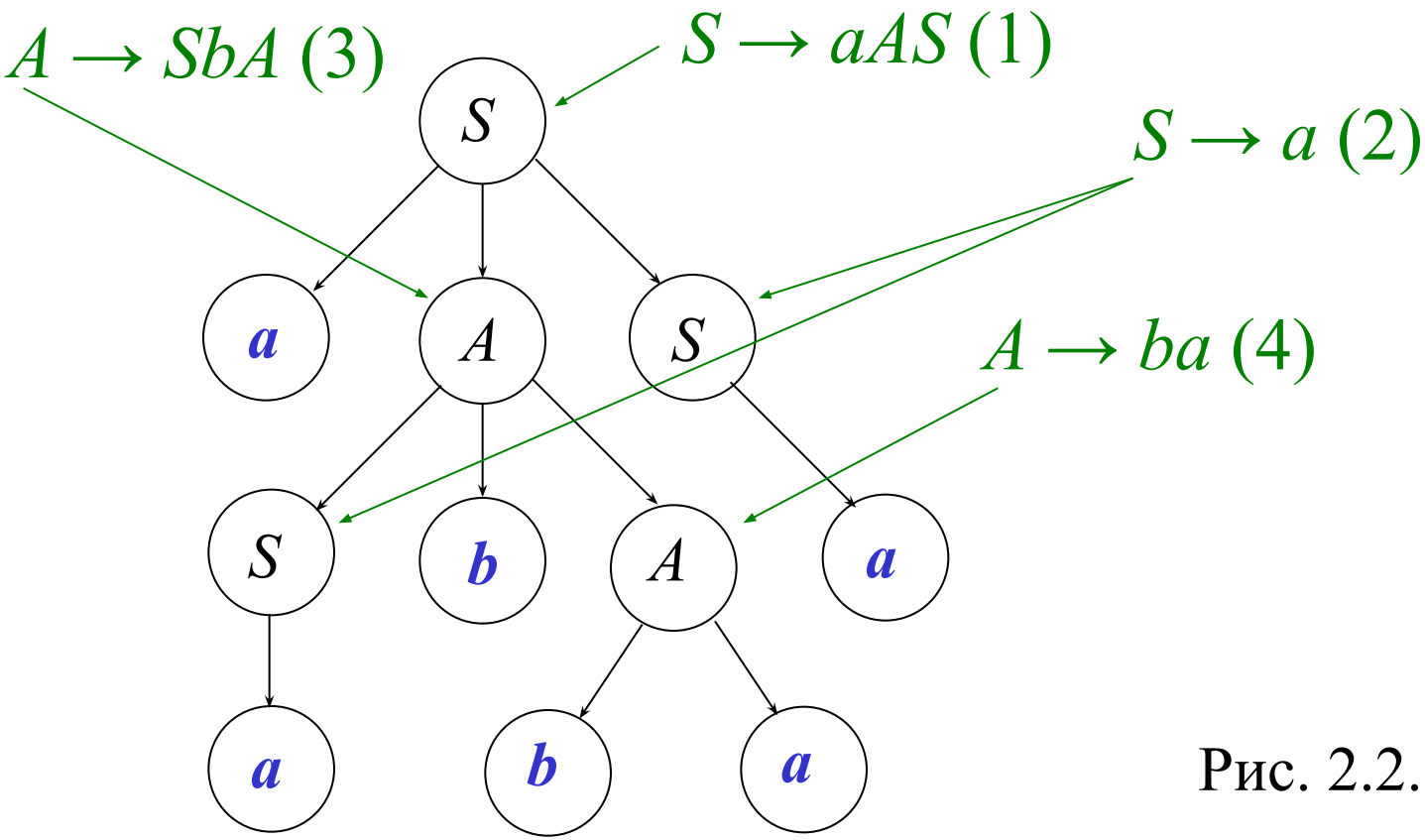

Рис. 2.2.

На рис. 2.2 изображено дерево, представляющее вывод:

$$S \overset{(1)}{\Rightarrow} aAS \overset{(3)}{\Rightarrow} aSbAS \overset{(2)}{\Rightarrow} aabAS \overset{(4)}{\Rightarrow} aabbaS \overset{(2)}{\Rightarrow} aabbaa.$$

Результат *aabbaa* этого дерева вывода получается, если выписать метки листьев слева направо.

Заметим, что в сентенциальных формах этого вывода на каждом шагу заменятся крайне левое вхождение нетерминала. Такие выводы в КС-грамматиках называются *левосторонними*.

Теорема 2.4. *Пусть* $G = (V_N, V_T, P, S)$ — *cfg*. *Вывод* $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$, *где* $\alpha \in V^*$, $\alpha \neq \varepsilon$, *существует тогда и только тогда, когда существует дерево вывода в грамматике* G *с результатом* α.

Доказательство. Будем доказывать аналогичное утверждение для грамматик $G_A = (V_N, V_T, P, A)$ с одними и теми же V_N, V_T и P, но с разными начальными символами $A \in V_N$.

Если это вспомогательное утверждение будет доказано для любой грамматики G_A, то справедливость утверждения теоремы будет следовать просто как частный случай при $A = S$.

Поскольку, как было сказано, во всех грамматиках одни и те же правила, то утверждение $A \overset{*}{\underset{G_A}{\Rightarrow}} \alpha$ эквивалентно утверждению $A \overset{*}{\underset{G_B}{\Rightarrow}} \alpha$ для любого $B \in V_{\mathrm{N}}$, в частности при $B = S$, то имеем также $A \overset{*}{\underset{G}{\Rightarrow}} \alpha$, т. к. $G_S = G$.

I. Пусть $\alpha \in V^+$ есть результат дерева вывода для грамматики G_A.

Индукцией по числу внутренних узлов k в дереве вывода покажем, что тогда $A \overset{*}{\underset{G_A}{\Rightarrow}} \alpha$.

База. Пусть $k = 1$, тогда имеется только один внутренний узел. В этом случае дерево имеет такой вид, как показано на рис. 2.3.

Retrun to slade 49

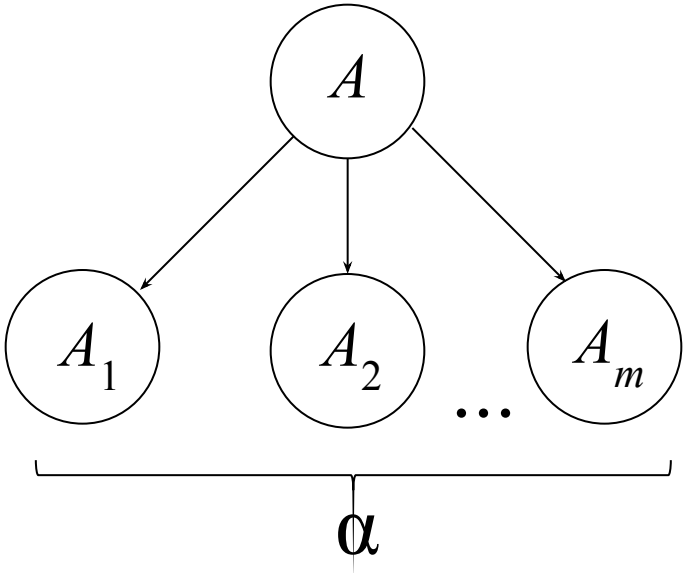

Рис. 2.3.

По определению дерева вывода

$$A \to A_1 A_2 \ldots A_m$$

должно быть правилом грамматики G_A и, следовательно, вывод $A \overset{*}{\underset{G_A}{\Rightarrow}} \alpha$ существует.

Return 97 Return 99 Return 102

Индукционная гипотеза. Предположим, что утверждение выполняется для всех $k \leq n$ ($n \geq 1$).

Индукционный переход. Пусть α есть результат дерева вывода с корнем, помеченным нетерминалом A, в котором k внутренних узлов, причем $k = n + 1$.

Рассмотрим прямых потомков корня данного дерева вывода (*см*. рис.2.3) . Они не могут быть все листьями, так как в противном случае дерево имело бы только одну внутреннюю вершину — корень, а их должно быть не меньше двух.

Перенумеруем эти узлы слева на право: 1, 2, ..., *m*. Если узел *i* — *не лист*, то он — корень некоторого поддерева, в котором внутренних узлов не больше *n*.

По индукционному предположению результат этого поддерева — обозначим его α_i — выводим из A_i, где A_i — конечно, нетерминал.

В обозначениях это можно записать так:

$$A_i \overset{*}{\underset{G_A}{\Rightarrow}} \alpha_i.$$

Если же A_i — *лист*, то $A_i = \alpha_i$ и в этом случае также $A_i \overset{*}{\underset{G_A}{\Rightarrow}} \alpha_i$.

Легко видеть, что если $i < j$, то узел i и все его потомки должны быть левее узла j и всех его потомков, и потому $\alpha = \alpha_1\alpha_2 \ldots \alpha_m$.

Мы можем теперь, используя правило $A \to A_1 A_2 \ldots A_m$ и все частичные выводы, следующие из индукционного предположения, выстроить вывод:

$$A \underset{G_A}{\Rightarrow} A_1 A_2 \ldots A_m \overset{*}{\underset{G_A}{\Rightarrow}} \alpha_1 A_2 \ldots A_m \overset{*}{\underset{G_A}{\Rightarrow}} \ldots \overset{*}{\underset{G_A}{\Rightarrow}} \alpha_1\alpha_2 \ldots A_m \overset{*}{\underset{G_A}{\Rightarrow}} \alpha_1\alpha_2\ldots\alpha_m = \alpha.$$

Итак, $A \overset{*}{\underset{G_A}{\Rightarrow}} \alpha$. Утверждение I доказано.

II. Пусть $A \overset{*}{\underset{G_A}{\Rightarrow}} \alpha$.

Индукцией по длине вывода l покажем, что существует дерево вывода в грамматике G_A, результат которого есть α.

База. Пусть $l = 1$.

Если $A \underset{G_A}{\Rightarrow} \alpha$, то на этом единственном шаге вывода используется правило $A \to \alpha \in P$.

Пусть $\alpha = A_1 A_2 \ldots A_m$, то по определению дерево, показанное на рис. 2.3, есть дерево вывода в грамматике G_A. Очевидно, что его результат есть α.

Индукционная гипотеза. Предположим, что утверждение выполняется для всех $l \le n$ ($n \ge 1$).

Индукционный переход. Пусть $A \overset{l}{\underset{G_A}{\Rightarrow}} \alpha$, где $l = n + 1$. Этот вывод имеет длину, по крайней мере, 2. Следовательно, имеется первый шаг и n других шагов ($n \ge 1$), т. е. вывод имеет вид

$$A \underset{G_A}{\Rightarrow} A_1 A_2 \ldots A_m \overset{l_1}{\underset{G_A}{\Rightarrow}} \alpha_1 A_2 \ldots A_m \overset{l_2}{\underset{G_A}{\Rightarrow}} \ldots \overset{l_m}{\underset{G_A}{\Rightarrow}} \alpha_1 \alpha_2 \ldots \alpha_m = \alpha.$$

Здесь $l_1 + l_2 + \ldots + l_m = n$, причем $l_i \le n$ ($1 \le i \le m$).

Если $l_i = 0$, то $\alpha_i = A_i$.

Если $l_i > 0$, то по индукционному предположению существует дерево вывода T_i с корнем, имеющем метку A_i и результатом α_i. Но первый шаг вывода предполагает существование правила

$$A \to A_1 A_2 \ldots A_m \in P.$$

Следовательно, можно построить дерево вывода, верхняя часть которого будет иметь вид, как на рис. 2. Следовательно, можно построить дерево вывода, верхняя часть которого будет иметь вид, как на рис. 2.3.

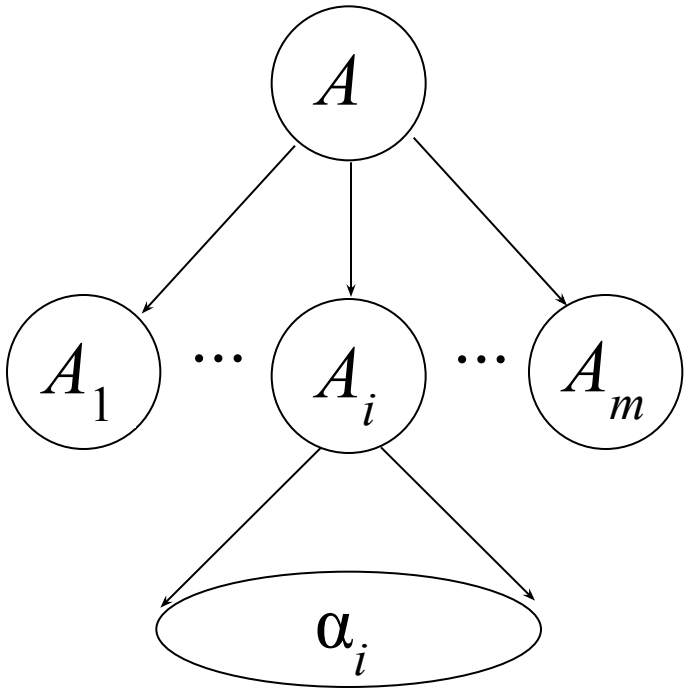

Рис. 2.4.

Далее, те вершины, которые помечены символами A_i и для которых существуют выводы вида $A_i \underset{G_A}{\overset{l_i}{\Rightarrow}} \alpha_i$ при $l_i > 0$, заменим деревьями вывода T_i с корнями, помеченными A_i, и результатами α_i. То, что получится (*см.* рис. 2.4), является деревом вывода $A \underset{G_A}{\overset{*}{\Rightarrow}} \alpha$ в грамматике G_A.

Утверждение II доказано. Из I и II при $A = S$ следует утверждение теоремы.