Глава 2

LL(*k*)-ГРАММАТИКИ И ТРАНСЛЯЦИИ

§ 2.1. Введение в *LL*(*k*)-грамматики

2.1.1. Неформальное описание

В гл. 1 было показано, что произвольная простая синтаксически управляемая трансляция всегда может быть реализована при помощи недетерминированного магазинного преобразователя. Если же, кроме того, схема, посредством которой задается эта трансляция, является семантически однозначной, то выход трансляции можно получить по левостороннему анализу входной цепочки при помощи детерминированного магазинного преобразователя. Следовательно, если найти такие классы входных грамматик, в которых левосторонний анализ может быть реализован детерминированным образом, то мы имели бы полностью детерминированную реализацию простых семантически однозначных трансляций.

Одним из таких подклассов КС-грамматик являются так называемые *LL*(*k*)-*грамматики*. Это самый большой "естественный" класс левоанализируемых грамматик.

***Определение* 2.1.** Пусть $\alpha = x\beta$ — левосентенциальная форма в некоторой cfg $G = (V_N, V_T, P, S)$, такая, что $x \in V_T^*$, а $\beta \in (V_N \cup V_T)^*$ начинается на нетерминал либо есть пустая цепочка. Цепочка x называется *закрытой частью*, а β — *открытой частью* левосентенциальной формы α.

Пример 2.1. Пусть $\alpha = abacAaB$. Закрытая часть α есть *abac*. Открытая ее часть — *AaB*. Если $\alpha = abc$, то закрытая часть есть *abc*, а открытая — ε.

Пусть $G = (V_N, V_T, P, S)$ — однозначная КС-грамматика и $w = a_1a_2\ldots a_n \in L(G)$. Тогда существует единственная последовательность левосентенциальных форм $\alpha_0, \alpha_1, \ldots, \alpha_m$, такая, что $S = \alpha_0$, $\alpha_i \underset{\text{lm}}{\overset{p_i}{\Rightarrow}} \alpha_{i+1}$ для $0 \le i < m$ и $\alpha_m = w$. Левосторонний анализ цепочки w есть $p_0, p_1, p_2, \ldots, p_{m-1}$.

Теперь предположим, что мы хотим найти этот левосторонний анализ, просматривая w слева направо один раз. Мы могли бы попытаться сделать это путем построения последовательности левосентенциальных форм $\alpha_0, \alpha_1, \ldots, \alpha_m$. Начальная сентенциальная форма $\alpha_0 = S$ уже известна. Остается определить способ построения следующей сентенциальной формы по последней из уже построенных.

Пусть $\alpha_i = a_1a_2\ldots a_jA\beta$ — последняя из построенных сентенциальных форм. Сравнивая закрытую часть этой сентенциальной формы $a_1a_2\ldots a_j$ с цепочкой w, мы можем определить ее окончание $a_{j+1}\ldots a_n$, вывод которого еще предстоит построить. Было бы желательно, чтобы α_{i+1} можно было найти, зная только $a_1a_2\ldots a_j$ — часть входной цепочки, которую мы уже просмотрели к этому мо-

менту, несколько следующих входных символов $a_{j+1}\dots a_{j+k}$ для некоторого фиксированного k и нетерминала A. Если эти три величины однозначно определяют, какое правило должно использоваться для раскрытия A, мы можем тогда точно определить α_{i+1} по α_i и k входным символам $a_{j+1}\dots a_{j+k}$.

Говорят, что грамматика, в которой *каждый* левосторонний вывод имеет это свойство, есть *LL*(k)-*грамматика*.

Будет показано, что для каждой *LL*(k)-грамматики можно построить детерминированный левосторонний анализатор, который действует за линейное время.

2.1.2. Формальное определение

***Определение* 2.2.** Пусть $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика. Определим функцию $\mathrm{FIRST}_k^G(\alpha) = \{w \in V_T^* \mid$ либо $|w| < k$ и $\alpha \overset{*}{\underset{G}{\Rightarrow}} w$, либо $|w| = k$ и $\alpha \overset{*}{\underset{G}{\Rightarrow}} wx$ для некоторой цепочки $x \in V_T^*\}$. Здесь $k \geq 0$ — целое, $\alpha \in (V_N \cup V_T)^*$.

Отметим, что эта функция определена, в частности, и для терминальной цепочки, и тогда, когда она пуста. При этом верхний индекс грамматики не существен, так как никакие правила для вывода терминальной цепочки из такого аргумента не требуются.

***Определение* 2.3.** Пусть $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика. Говорят, что G есть *LL*(k)-*грамматика* для некоторого фиксированного k, если для любых двух левосторонних выводов вида

1) $S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx$,

2) $S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} wy$,

в которых $\mathrm{FIRST}_k^G(x) = \mathrm{FIRST}_k^G(y)$, имеет место равенство $\beta = \gamma$.

***Определение* 2.4.** Говорят, что контекстно-свободная грамматика G есть *LL-грамматика*, если она *LL*(k) для некоторого $k \geq 0$.

Пример 2.2. Пусть $G = (\{S, B\}, \{a, b\}, P, S)$, где $P = \{S \to aBS \mid b, B \to a \mid bSB\}$. По определению грамматика G — *LL*(1), если

1) $S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx$,

2) $S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} wy$,

и если x и y начинаются на один и тот же символ, то должно быть $\beta = \gamma$.

Рассмотрим два левосторонних вывода, в которых роль нетерминала A играет символ S.

Имеем (1) $S \underset{lm}{\Rightarrow} aBS$ и (2) $S \underset{lm}{\Rightarrow} b$. Тогда $w = \alpha = \varepsilon$, $\beta = aBS$, $\gamma = b$. Ясно, что любая цепочка x, выводимая из $\beta\alpha = aBS$, начинается на a, а цепочка y, выводимая из $\gamma\alpha = b$, равна b. Поэтому если первый символ цепочки, следующей за закрытой частью сентенциальной формы ($w = \varepsilon$), есть a, то для замены нетерминала S следует использовать первую альтернативу. Если она начинается на b, то вторую.

Рассмотрим два левосторонних вывода, в которых роль нетерминала A играет символ B:

Имеем (1) $S \underset{lm}{\Rightarrow} aBS \underset{lm}{\Rightarrow} aaS$ и (2) $S \underset{lm}{\Rightarrow} aBS \underset{lm}{\Rightarrow} abSBS$. Тогда $w = a$, $\alpha = S$, $\beta = a$, $\gamma = bSB$. Ясно, что любая цепочка x, выводимая из $\beta\alpha = aS$, начинается на a, а цепочка y, выводимая из $\gamma\alpha = bSBS$, начинается на b. Поэтому если первый символ цепочки, следующей за закрытой частью сентенциальной формы ($w = a$), есть a, то для замены нетерминала B следует использовать первую альтернативу. Если она начинается на b, то вторую.

В обоих случаях $LL(1)$-условие выполнено: по первому символу цепочки, следующей за закрытой частью сентенциальной формы, однозначно определяется то правило, которое следует применить к соответствующему нетерминалу, чтобы в конце концов получить анализируемую цепочку. Очевидно, что любые два левосторонних вывода в данной грамматике, подпадающие под вышеприведенный образец, удовлетворяют $LL(k)$-условию.

Данная грамматика служит примером *простой* $LL(1)$-грамматики.

***Определение* 2.5.** Говорят, что контекстно-свободная грамматика G является *простой* $LL(1)$-*грамматикой*, если в ней нет ε-правил, и все альтернативы для каждого нетерминала начинаются с терминалов и притом различных.

Таким образом, в простой $LL(1)$-грамматике для данной пары (A, a), где $A \in V_N$ и $a \in V_T$, существует самое большее — одна альтернатива вида $A \to a\alpha$.

§ 2.2. Свойства *LL*(*k*)-грамматик

Теорема 2.1. *Чтобы контекстно-свободная грамматика* $G = (V_N, V_T, P, S)$ *была* $LL(k)$-*грамматикой, необходимо и достаточно, чтобы*

$$\mathrm{FIRST}_k^G(\beta\alpha) \cap \mathrm{FIRST}_k^G(\gamma\alpha) = \varnothing$$

для всех α, β, γ, *таких, что существуют правила* $A \to \beta$, $A \to \gamma \in P$, $\beta \neq \gamma$ *и существует вывод* $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$.

Доказательство. Будем предполагать, что грамматика G не содержит бесполезных нетерминалов. Это предположение не умаляет общности рассуждений, так как бесполезные нетерминалы не влияют на $LL(k)$-условие, фигурирующее в определении $LL(k)$-грамматик. Обе части доказательства проведем методом “от противного”.

I. *Необходимость*. Пусть G — $LL(k)$-грамматика, но условие не выполнено, т.е. нашлись такие цепочки α, β, γ, что существуют $A \to \beta$, $A \to \gamma \in P$, $\beta \neq \gamma$ и существует вывод $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$, для которых $\mathrm{FIRST}_k^G(\beta\alpha) \cap \mathrm{FIRST}_k^G(\gamma\alpha) \neq \varnothing$.

Пусть некоторая цепочка $z \in \mathrm{FIRST}_k^G(\beta\alpha) \cap \mathrm{FIRST}_k^G(\gamma\alpha)$. Мы можем продолжить имеющийся вывод $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$ двумя способами, используя упомянутые два правила и учитывая определение функции FIRST_k^G:

1) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} wzu$,

2) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} wzv$

для некоторых $u, v \in V_T^*$. При построении этих выводов была использована теорема о том, что всякий вывод терминальной цепочки может быть перестроен в левосторонний.

Остается сопоставить эти два левосторонних вывода с теми, которые участвуют в определении $LL(k)$-грамматик. Здесь в роли цепочки x используется zu, а в роли цепочки y — zv. Очевидно, $\mathrm{FIRST}_k^G(x) = \mathrm{FIRST}_k^G(y) = z$, но $\beta \neq \gamma$, что противоречит предположению о том, что G — $LL(k)$-грамматика.

В частности, если $|z| < k$ и $u = v = \varepsilon$, то имеем два разных левосторонних вывода одной и той же терминальной цепочки wz. Это означает, что G — не однозначная грамматика и, естественно, не $LL(k)$ ни при каком k. Необходимость доказана.

II. *Достаточность*. Пусть условие теоремы выполнено, но G — не $LL(k)$-грамматика. Это значит, что существуют два левосторонних вывода:

1) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} wx$,

2) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} wy$,

в которых $\mathrm{FIRST}_k^G(x) = \mathrm{FIRST}_k^G(y)$, но $\beta \neq \gamma$.

Пусть $\mathrm{FIRST}_k^G(x) = \mathrm{FIRST}_k^G(y) = z$. Имеем $\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} x$ и $\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} y$. Согласно определению функции FIRST_k^G из существования этих двух выводов заключаем, что

$z = \mathrm{FIRST}_k^G(x) \in \mathrm{FIRST}_k^G(\beta\alpha)$,

$z = \mathrm{FIRST}_k^G(y) \in \mathrm{FIRST}_k^G(\gamma\alpha)$

и, следовательно,

$z \in \mathrm{FIRST}_k^G(\beta\alpha) \cap \mathrm{FIRST}_k^G(\gamma\alpha) \neq \varnothing$,

что входит в противоречие с первоначальным предположением о том, что условие теоремы выполняется. Достаточность доказана, а вместе с ней и вся теорема.

***Определение* 2.6.** Пусть $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика, и $\beta \in V^*$. Определим *функцию* $\mathrm{FOLLOW}_k^G(\beta) = \{w \in V_T^* \mid S \underset{G}{\overset{*}{\Rightarrow}} \gamma\beta\alpha$ и $w \in \mathrm{FIRST}_k^G(\alpha)\}$. Здесь $k \geq 0$ — целое.

Теорема 2.2. *Чтобы контекстно-свободная грамматика* $G = (V_N, V_T, P, S)$ *была* $LL(1)$*-грамматикой, необходимо и достаточно, чтобы*

$$\mathrm{FIRST}_1^G(\beta\,\mathrm{FOLLOW}_1^G(A)) \cap \mathrm{FIRST}_1^G(\gamma\,\mathrm{FOLLOW}_1^G(A)) = \varnothing$$

для всех $A \in V_N$, $\beta, \gamma \in (V_N \cup V_T)^*$, *таких, что существуют правила* $A \to \beta$, $A \to \gamma \in P$, $\beta \neq \gamma$.

Доказательство. Требуется пояснить, что в формулировке условия теоремы аргумент функции FIRST_1^G — не цепочка, как было определено ранее, а множе-

ство цепочек. Расширение области определения этой функции на множества производится естественным образом:

$$\mathrm{FIRST}_1^G(W) = \bigcup_{s \in W} \mathrm{FIRST}_1^G(s),$$

где $W \subseteq V^*$.

Обе части теоремы доказываются методом “от противного”. Как и ранее, не нарушая общности рассуждений, будем считать грамматику G приведенной.

I. *Необходимость*. Пусть G — $LL(1)$-грамматика, но условие не выполнено, т.е. существуют правила $A \to \beta,\ A \to \gamma \in P,\ \beta \neq \gamma,\ c \in (V_\mathrm{T} \cup \{\varepsilon\})$, такие, что

$$c \in \mathrm{FIRST}_1^G(\beta\,\mathrm{FOLLOW}_1^G(A)) \text{ и } c \in \mathrm{FIRST}_1^G(\gamma\,\mathrm{FOLLOW}_1^G(A)).$$

Это означает, что существуют $a, b \in \mathrm{FOLLOW}_1^G(A)$, такие, что $c \in \mathrm{FIRST}_1^G(\beta a)$ и $c \in \mathrm{FIRST}_1^G(\gamma b)$, и согласно определению функции FIRST_1^G существуют выводы:

$$\beta a \overset{*}{\underset{G}{\Rightarrow}} cu \text{ и } \gamma v \overset{*}{\underset{G}{\Rightarrow}} cv, \text{ если } c \in V_\mathrm{T}$$

или

$$\beta a \overset{*}{\underset{G}{\Rightarrow}} \varepsilon \text{ и } \gamma b \overset{*}{\underset{G}{\Rightarrow}} \varepsilon, \text{ если } c = \varepsilon.$$

Случай 1: $c \in V_\mathrm{T}$, $\beta a \overset{*}{\underset{G}{\Rightarrow}} cu\underbrace{a}_{u} = cu$, $\beta \overset{*}{\underset{G}{\Rightarrow}} cu'$, $u' \in V_\mathrm{T}^*$; $\gamma b \overset{*}{\underset{G}{\Rightarrow}} cv'\underbrace{b}_{v}$, $\gamma \overset{*}{\underset{G}{\Rightarrow}} cv'$, $v' \in V_\mathrm{T}^*$.

Так как грамматика G — приведенная, то существует вывод, в частности левосторонний, который порождает сентенциальную форму, содержащую A: $S \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wA\alpha$, который может быть продолжен двумя способами:

1) $S \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wA\alpha \underset{\mathrm{lm}}{\Rightarrow} w\beta\alpha \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wcu'\alpha \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} w\underbrace{cu'z}_{x}$,

2) $S \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wA\alpha \underset{\mathrm{lm}}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wcv'\alpha \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} w\underbrace{cv'z}_{y}$,

где вывод $\alpha \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} z$, $z \in V_\mathrm{T}^*$, существует также в силу приведенности грамматики G.

Итак, имеем два левосторонних вывода (1 и 2), фигурирующие в определении $LL(k)$-грамматик, в которых в роли цепочки x используется $cu'z$, в роли цепочки y — $cv'z$, причем

$$\mathrm{FIRST}_1^G(x) = \mathrm{FIRST}_1^G(cu'z) = c,\ \mathrm{FIRST}_1^G(y) = \mathrm{FIRST}_1^G(cv'z) = c,$$

но $\beta \neq \gamma$. Следовательно, грамматика G — не $LL(1)$, что противоречит первоначальному предположению.

Случай 2: $c \in V_\mathrm{T}$, $\beta \overset{*}{\underset{G}{\Rightarrow}} \varepsilon$, $a = c$, $a \in \mathrm{FOLLOW}_1^G(A)$; $\gamma b \overset{*}{\underset{G}{\Rightarrow}} c\underbrace{v'b}_{v}$, $\gamma \overset{*}{\underset{G}{\Rightarrow}} cv'$, $v' \in V_\mathrm{T}^*$.

Согласно определению $c \in \mathrm{FOLLOW}_1^G(A)$ означает, что существует вывод, в частности левосторонний, вида $S \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wA\alpha$, такой, что $c \in \mathrm{FIRST}_1^G(\alpha)$ или, что то же самое, существует вывод $\alpha \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} cz$ при некотором $z \in V_\mathrm{T}^*$. Продолжим левосторонним образом вывод $S \overset{*}{\underset{\mathrm{lm}}{\Rightarrow}} wA\alpha$ двумя способами:

1) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\underbrace{cz}_{x}$,

2) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} wcv'\alpha \underset{lm}{\overset{*}{\Rightarrow}} wcv'cz$.

Получили два вывода (1 и 2), фигурирующие в определении $LL(k)$-грамматик, с cz в роли цепочки x и $cv'cz$ в роли цепочки y. Имеем $\mathrm{FIRST}_1^G(x) = \mathrm{FIRST}_1^G(cz) = c$, $\mathrm{FIRST}_1^G(y) = \mathrm{FIRST}_1^G(cv'cz) = c$, т.е. $\mathrm{FIRST}_1^G(x) = \mathrm{FIRST}_1^G(y)$, но $\beta \neq \gamma$. Следовательно, грамматика G — не $LL(1)$, что противоречит первоначальному предположению.

Случай 3: $c \in V_T$, $\beta a \underset{G}{\overset{*}{\Rightarrow}} cu'a$, $\beta \underset{G}{\overset{*}{\Rightarrow}} cu'$, $u' \in V_T^*$; $\gamma \underset{G}{\overset{*}{\Rightarrow}} \varepsilon$, $b = c$, $b \in \mathrm{FOLLOW}_1^G(A)$.

Он разбирается аналогично предыдущему.

Случай 4: $c \in V_T$, $\beta \underset{G}{\overset{*}{\Rightarrow}} \varepsilon$, $\gamma \underset{G}{\overset{*}{\Rightarrow}} \varepsilon$, $a = b = c$, $c \in \mathrm{FOLLOW}_1^G(A)$.

Согласно определению $c \in \mathrm{FOLLOW}_1^G(A)$ означает существование вывода вида $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$, такого, что $c \in \mathrm{FIRST}_1^G(\alpha)$ или, что то же самое, $\alpha \underset{lm}{\overset{*}{\Rightarrow}} cz$ при некотором $z \in V_T^*$.

Теперь вывод $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$ продолжим левосторонним образом двумя способами:

1) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\underbrace{cz}_{x}$,

2) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\underbrace{cz}_{y}$,

Мы получили два левосторонних вывода (1 и 2), фигурирующих в определении $LL(k)$-грамматик, в которых в роли цепочек x и y используются одинаковые терминальные цепочки cz, функция FIRST_1^G от них дает одинаковый результат, равный c, притом что $\beta \neq \gamma$. Следовательно, грамматика G — не $LL(1)$, что противоречит первоначальному предположению[12].

Случай 5: $\beta \underset{G}{\overset{*}{\Rightarrow}} \varepsilon$, $\gamma \underset{G}{\overset{*}{\Rightarrow}} \varepsilon$, $a = b = \varepsilon$, $\varepsilon \in \mathrm{FOLLOW}_1^G(A)$.

Как и в случае 4, $\varepsilon \in \mathrm{FOLLOW}_1^G(A)$ означает существование вывода $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$, такого, что $\alpha \underset{lm}{\overset{*}{\Rightarrow}} \varepsilon$. Его можно продолжить левосторонним образом двумя способами:

1) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\alpha \underset{lm}{\overset{*}{\Rightarrow}} w$,

2) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} w\alpha \underset{lm}{\overset{*}{\Rightarrow}} w$.

[12] Противоречие можно видеть и в том, что существуют два разных левосторонних вывода для одной и той же терминальной цепочки az.

В роли цепочек x и y здесь используется ε. Имеем $\mathrm{FIRST}_1^G(x) = \mathrm{FIRST}_1^G(y) = \varepsilon$, хотя по-прежнему $\beta \neq \gamma$. Следовательно, грамматика G — не $LL(1)$, что противоречит первоначальному предположению[13]. Необходимость доказана.

II. *Достаточность*. Пусть условие теоремы выполнено, но грамматика G — не $LL(1)$. Тогда существуют два левосторонних вывода вида

1) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} wx$,

2) $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} wy$,

в которых $\mathrm{FIRST}_1^G(x) = \mathrm{FIRST}_1^G(y)$, но $\beta \neq \gamma$. Поскольку $\beta\alpha \underset{lm}{\overset{*}{\Rightarrow}} x$ и $\gamma\alpha \underset{lm}{\overset{*}{\Rightarrow}} y$, то

$$\mathrm{FIRST}_1^G(x) \subseteq \mathrm{FIRST}_1^G(\beta\alpha)$$

$$\mathrm{FIRST}_1^G(y) \subseteq \mathrm{FIRST}_1^G(\gamma\alpha)$$

и тогда заключаем, что

$$\mathrm{FIRST}_1^G(x) = \mathrm{FIRST}_1^G(y) \subseteq \mathrm{FIRST}_1^G(\beta\alpha) \cap \mathrm{FIRST}_1^G(\gamma\alpha) \neq \varnothing,$$

что противоречит первоначальному предположению о том, что условие теоремы выполнено. Достаточность и теорема доказаны.

Следствие 2.1. Теорему 2.2 можно переформулировать следующим образом: КС-грамматика G является LL(1)-грамматикой тогда и только тогда, когда для каждого множества A-правил: $A \to \alpha_1 | \alpha_2 | \ldots | \alpha_n$ — выполняются следующие условия:

1) $\mathrm{FIRST}_1^G(\alpha_i) \cap \mathrm{FIRST}_1^G(\alpha_j) = \varnothing$ *при* $i \neq j$, $1 \le i, j \le n$;

2) *если* $\alpha_i \underset{G}{\overset{*}{\Rightarrow}} \varepsilon$, *то* $\mathrm{FIRST}_1^G(\alpha_j) \cap \mathrm{FOLLOW}_1^G(A) = \varnothing$ *для всех* j: $1 \le j \le n$, $j \neq i$.

***Определение* 2.7.** Контекстно-свободная грамматика $G = (V_N, V_T, P, S)$ называется *сильной LL(k)-грамматикой*, если она удовлетворяет условию теоремы 2.2:

$$\mathrm{FIRST}_k^G(\beta\,\mathrm{FOLLOW}_k^G(A)) \cap \mathrm{FIRST}_k^G(\gamma\,\mathrm{FOLLOW}_k^G(A)) = \varnothing$$

для всех $A \in V_N$, $\beta, \gamma \in (V_N \cup V_T)^*$, таких, что существуют правила $A \to \beta$, $A \to \gamma \in P$, $\beta \neq \gamma$.

***Следствие* 2.2.** Каждая $LL(1)$-грамматика является сильной. Однако следующий пример показывает, что для $k > 1$ не всякая $LL(k)$-грамматика является сильной.

Пример 2.3. Рассмотрим cfg $G = (\{S, A\}, \{a, b\}, P, S)$, где

$P = \{S \to aAaa \mid bAba, A \to b \mid \varepsilon\}$.

Используя теорему 2.1, проверим, что грамматика G — $LL(2)$.

---

[13] Противоречие можно видеть и в том, что существуют два разных левосторонних вывода для одной и той же терминальной цепочки w.

I. Проведем тестирование относительно нетерминала S:

$\mathrm{FIRST}_2^G(aAaa\alpha) = \{aa, ab\}$,

$\mathrm{FIRST}_2^G(bAba\alpha) = \{bb\}$ независимо от α,

$\mathrm{FIRST}_2^G(aAaa\alpha) \cap \mathrm{FIRST}_2^G(bAba\alpha) = \{aa, ab\} \cap \{bb\} = \varnothing$.

II. Проведем тестирование относительно нетерминала A. При этом следует учесть, что $S \Rightarrow aAaa$ и $S \Rightarrow bAba$, так что $\alpha \in \{aa, ba\}$. Тогда при $\alpha = aa$

$\mathrm{FIRST}_2^G(baa) = \{baa\}$,

$\mathrm{FIRST}_2^G(aa) = \{aa\}$,

$\mathrm{FIRST}_2^G(baa) \cap \mathrm{FIRST}_2^G(aa) = \{baa\} \cap \{aa\} = \varnothing$;

при $\alpha = ba$

$\mathrm{FIRST}_2^G(bba) = \{bb\}$,

$\mathrm{FIRST}_2^G(ba) = \{ba\}$,

$\mathrm{FIRST}_2^G(bba) \cap \mathrm{FIRST}_2^G(ba) = \{bb\} \cap \{ba\} = \varnothing$.

Так что G — действительно $LL(2)$-грамматика.

Но поскольку, очевидно, что $\mathrm{FOLLOW}_2^G(S) = \{\varepsilon\}$, то, учитывая существование выводов $S \Rightarrow aAaa$ и $S \Rightarrow bAba$, заключаем, что $\mathrm{FOLLOW}_2^G(A) = \{aa, ba\}$.

Проверка условия сильной $LL(2)$-грамматики показывает, что для S

$\mathrm{FIRST}_2^G(aAaa\{\varepsilon\}) \cap \mathrm{FIRST}_2^G(bAba\{\varepsilon\}) = \{aa, ab\} \cap \{bb\} = \varnothing$,

а для A

$\mathrm{FIRST}_2^G(b\{aa, ba\}) \cap \mathrm{FIRST}_2^G(\varepsilon\{aa, ba\}) = \{ba, bb\} \cap \{aa, ba\} = \{ba\} \neq \varnothing$.

Таким образом, грамматика G — $LL(2)$, но не сильная $LL(2)$-грамматика.

Теорема 2.3. *Если $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика, и G — леворекурсивна, то G — не $LL(k)$-грамматика ни при каком k.*

Доказательство будем проводить в предположении приведенности грамматики G, поскольку, как отмечалось ранее, бесполезные нетерминалы не влияют на выполнение $LL(k)$-критерия.

Так как грамматика G леворекурсивна, то существует вывод вида $A \overset{+}{\underset{G}{\Rightarrow}} A\rho$, где $\rho \neq \varepsilon$, хотя и не исключается, что $\rho \overset{*}{\underset{G}{\Rightarrow}} \varepsilon$. Напомним, что $A \overset{+}{\underset{G}{\Rightarrow}} A\rho$ означает, что существует вывод вида $A \underset{G}{\Rightarrow} \beta \overset{*}{\underset{G}{\Rightarrow}} A\rho$.

Отметим, что независимо от того, $\beta = A\rho$ или $\beta \neq A\rho$, должно существовать еще какое-то правило для A, скажем, $A \to \gamma$ ($\gamma \neq \beta$), ибо в противном случае ни-

чего, кроме $A \overset{*}{\underset{G}{\Rightarrow}} A\rho^i$, где $i \geq 0$, вывести из нетерминала A было бы невозможно, что противоречило бы приведенности грамматики G.

В любом случае вывод $A \overset{+}{\underset{G}{\Rightarrow}} A\rho$ в общем случае состоит из шагов вида

$$A \underset{G}{\Rightarrow} A_1\alpha_1 \underset{G}{\Rightarrow} A_2\alpha_2\alpha_1 \underset{G}{\Rightarrow} \ldots \underset{G}{\Rightarrow} A_{m-1}\alpha_{m-1}\ldots\alpha_1 \underset{G}{\Rightarrow} A_m\alpha_m\ldots\alpha_1 = A\rho,$$

где $A_m = A$, $\rho = \alpha_m\ldots\alpha_1$. Очевидно, что в этом выводе были использованы правила $A \to A_1\alpha_1$, $A_1 \to A_2\alpha_2, \ldots$, $A_{m-1} \to A_m\alpha_m$, где $A_m = A$. Хотя бы одно из этих правил должно иметь альтернативу, не начинающуюся ни на один из нетерминалов A_1, $A_2, \ldots$, A_m (иначе грамматика G была бы неприведенной). Пусть, например, существует $A_j \to A_{j+1}\alpha_{j+1} \mid \gamma$, где $\gamma \neq A_{j+1}\alpha_{j+1}$.

Так как грамматика G — приведенная, существует сентенциальная форма, в частности левостороннего вывода, в которой участвует нетерминал A, например $S \overset{*}{\underset{\text{lm}}{\Rightarrow}} wA\alpha$, который может быть продолжен следующим образом:

$$S \overset{*}{\underset{\text{lm}}{\Rightarrow}} wA\alpha \overset{*}{\underset{\text{lm}}{\Rightarrow}} wA\rho^i\alpha \underset{\text{lm}}{\Rightarrow} wA_1\alpha_1\rho^i\alpha \underset{\text{lm}}{\Rightarrow} \ldots \underset{\text{lm}}{\Rightarrow} wA_j\alpha_j\ldots\alpha_1\rho^i\alpha.$$

Далее этот вывод можно продолжить двумя способами:

$$1) \underset{\text{lm}}{\Rightarrow} wA_{j+1}\alpha_{j+1}\alpha_j\ldots\alpha_1\rho^i\alpha \overset{*}{\underset{\text{lm}}{\Rightarrow}} wA_m\alpha_m\ldots\alpha_1\rho^i\alpha = wA\rho^{i+1}\alpha \overset{*}{\underset{\text{lm}}{\Rightarrow}}$$

$$\overset{*}{\underset{\text{lm}}{\Rightarrow}} wA_j\alpha_j\ldots\alpha_1\rho^{i+1}\alpha \underset{\text{lm}}{\Rightarrow} w\gamma\alpha_j\ldots\alpha_1\rho^{i+1}\alpha \overset{*}{\underset{\text{lm}}{\Rightarrow}} w\underbrace{zx_j\ldots x_1r^{i+1}}_{x}t,$$

$$2) \underset{\text{lm}}{\Rightarrow} w\gamma\alpha_j\ldots\alpha_1\rho^i\alpha \overset{*}{\underset{\text{lm}}{\Rightarrow}} w\underbrace{zx_j\ldots x_1r^i}_{y}t,$$

предполагая, что ввиду приведенности грамматики существуют выводы

$$\gamma \overset{*}{\underset{\text{lm}}{\Rightarrow}} z,\ \alpha_j \overset{*}{\underset{\text{lm}}{\Rightarrow}} x_j,\ldots,\ \alpha_1 \overset{*}{\underset{\text{lm}}{\Rightarrow}} x_1,\ \rho \overset{*}{\underset{\text{lm}}{\Rightarrow}} r,\ \alpha \overset{*}{\underset{\text{lm}}{\Rightarrow}} t, \text{ где } w,z,x_j,\ldots,x_1,r,t \in V_{\text{T}}^*.$$

Эти два вывода можно сопоставить с теми, которые фигурируют в определении $LL(k)$-грамматик. Здесь $x = zx_j\ldots x_1r^{i+1}t$, а $y = wzx_j\ldots x_1r^it$.

Если $r \neq \varepsilon$, то каким бы большим ни было k, всегда путем выбора i можно добиться, чтобы $\text{FIRST}_k^G(x) = \text{FIRST}_k^G(y)$ при том, что в точке, где эти два вывода разошлись, были применены различные альтернативы для раскрытия нетерминала A_j:

$$A_j \to A_{j+1}\alpha_{j+1} \mid \gamma,\ \gamma \neq A_{j+1}\alpha_{j+1}.$$

Если $r = \varepsilon$, то имеем два разных левосторонних вывода одной и той же терминальной цепочки $wzx_j\ldots x_1t$. То и другое означает, что грамматика G — не $LL(k)$ ни при каком k.

***Следствие* 2.3.** Итак, мы имеем два достаточных признака для того, чтобы считать КС-грамматику не LL-грамматикой. Это — *неоднозначность* и *леворекурсивность*.

§ 2.3. *k*-Предсказывающие алгоритмы анализа

Для класса *LL*(*k*)-грамматик существует адекватный класс анализаторов, называемых *k-предсказывающими алгоритмами анализа*.

2.3.1. Неформальное описание

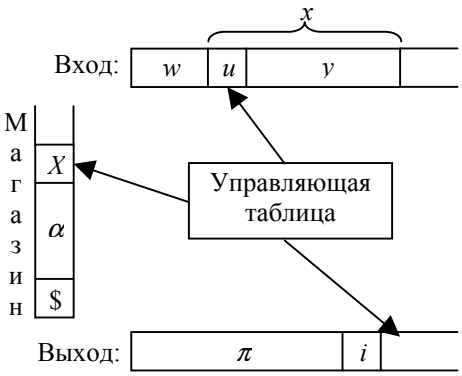

Рис. 2.1.

Это устройство (рис. 2.1) имеет разбитые на ячейки входную, выходную ленты и ленту магазина. “Дно” магазина маркируется специальным символом, например $, который постоянно находится на магазинной ленте. Конечное устройство управления руководствуется управляющей таблицей, создаваемой по грамматике, в которой производится синтаксический анализ. Она определяет действия в зависимости от *k* символов, сканируемых единовременно читающей головкой входной ленты — эта часть входной цепочки называется *аванцепочкой*, и верхнего символа магазина. Этими параметрами определяются движения следующих двух типов:

1. Запись вместо верхнего символа магазина некоторой цепочки магазинных символов и выдача на выходную ленту некоторой цепочки выходных символов. Выходная головка при этом продвигается к ближайшей свободной ячейке. В случае использования этого устройства для целей анализа на выходную ленту записываются номера правил, образующих левосторонний анализ входной цепочки, если она принимается. Этот тип движений не продвигает входную головку.

2. Сброс верхнего символа магазина и продвижение читающей головки вправо к следующей ячейке входной ленты. Такие движения происходят только тогда, когда первый символ аванцепочки — такой же, как верхний символ магазина. При этих движениях на выходную ленту ничего не пишется.

Существует возможность сигнализации о приеме входной цепочки или об ошибке в ней.

k-Предсказывающий алгоритм анализа напоминает детерминированный магазинный преобразователь, но отличается от него тем, что “видит” сразу *k* символов на входе и не имеет внутренних состояний управления. В начальный момент на входе находится анализируемая цепочка, в магазине — начальный сим-

вол магазина и маркер его "дна", на выходе пусто. Читающая головка сканирует начало входной ленты. На каждом такте работы алгоритм адресуется к своей управляющей таблице с двумя входами: верхним символом магазина и текущей аванцепочкой. Управляющий элемент диктует одно движение из двух, которое и выполняется. Этот цикл повторяется до тех пор, пока управляющий элемент не просигнализирует о приеме входной цепочки, и в этом случае на выходе образуется ее левосторонний анализ, или управляющий элемент диагностирует ошибку на входе, и тогда алгоритм останавливается, не принимая входной цепочки.

На рис. 2.1 входная цепочка представлена как wx, причем w обозначает просмотренную, а x — не просмотренную ее часть, которая начинается с аванцепочки u и заканчивается цепочкой y. Магазинная цепочка представлена в виде $X\alpha\$$, где X обозначает верхний символ магазина, α — символы магазина, располагающиеся ниже его вершины, а \$ — маркер "дна". Выходная цепочка представлена как πi, где π обозначает часть цепочки, образованную перед последним движением алгоритма, а i — последнюю произведенную запись на выходную ленту.

2.3.2. Формальное определение.

Сначала дадим несколько определений.

***Определение* 2.8.** *k-Предсказывающим алгоритмом анализа* называется формальная система $\mathfrak{A} = (\Sigma,\ \Gamma \cup \{\$\},\ \Delta,\ M,\ X_0,\ \$)$, где Σ — *входной алфавит*, $\Gamma \cup \{\$\}$ — *магазинный алфавит*, $\$ \notin \Gamma$ — *маркер* "*дна*" *магазина*, Δ — *выходной алфавит*, $X_0 \in \Gamma$ — *начальный символ магазина*, M: $(\Gamma \cup \{\$\}) \times \Sigma^{*k} \to \{(\beta, i), \text{pop}, \text{accept}, \text{error}\}$ — *управляющая таблица*, $\beta \in \Gamma^*$, $i \in \Delta$ — номер правила грамматики.

Работу k-предсказывающего алгоритма анализа проще всего описать в терминах отношения $\vdash$ на множестве конфигураций.

***Определение* 2.9.** Под *конфигурацией* k-предсказывающего алгоритма анализа будем подразумевать тройку (x, α, π), где $x \in \Sigma^*$ — непросмотренная часть входной цепочки, причем $u = \text{FIRST}_k(x) \in \Sigma^{*k}$ — аванцепочка, $\alpha \in \Gamma^*\{\$\}$ — магазинная цепочка, $\pi \in \Delta^*$ — выходная цепочка.

Начальная конфигурация есть $(w, X_0\$, \varepsilon)$, где $w \in \Sigma^*$ — вся входная цепочка.

Пусть $(x, X\alpha, \pi)$ — текущая конфигурация, где $x \in \Sigma^*$ — непросмотренная часть входной цепочки, $X \in \Gamma \cup \{\varepsilon\}$ — верхний символ магазина или пустая цепочка, $\alpha \in \Gamma^*\{\$\}$ — остальная часть магазинной цепочки.

Определим следующую конфигурацию в зависимости от значения элемента управляющей таблицы $M(X, u)$.

1. Если $M(X, u) = (\beta, i)$, то $(x, X\alpha, \pi) \vdash (x, \beta\alpha, \pi i)$.

2. Если $M(X, u) = \text{pop}$ и в этом случае всегда $X = a$, $a \in \Sigma$, $x = ax'$, $x' \in \Sigma^*$, то $(x, X\alpha, \pi) = (ax',\ a\alpha, \pi) \vdash (x', \alpha, \pi)$.

3. Если $M(X, u)$ = accept, что бывает только по достижении *конечной* или *принимающей конфигурации* $(\varepsilon, \$, \pi)$, то анализатор *останавливается*, *принимая входную цепочку*.

4. Если M(X, u) = error, то анализатор сообщает об ошибке на входе и останавливается, не принимая входной цепочки.

Как обычно, определяется степень ($\vdash^n$), транзитивное замыкание ($\vdash^+$) и рефлексивно-транзитивное замыкание ($\vdash^*$) этого отношения на конфигурациях.

Если $(w, X_0\$, \varepsilon) \vdash^* (\varepsilon, \$, \pi)$, то мы пишем $\mathfrak{A}(w) = \pi$ и называем π *выходом* $\mathfrak{A}$ для *входа* w.

Если из начальной конфигурации $(w, X_0\$, \varepsilon)$ алгоритм $\mathfrak{A}$ не достигает принимающей конфигурации, то говорят, что значение $\mathfrak{A}(w)$ *не определено*.

Трансляция, определяемая k-предсказывающим алгоритмом анализа $\mathfrak{A}$, есть $\tau(\mathfrak{A}) = \{(w, \pi) \mid \mathfrak{A}(w) = \pi\}$.

Говорят, что $\mathfrak{A}$ есть *правильный* k-предсказывающий алгоритм анализа для cfg G, если (1) $L(G) = \{w \mid \mathfrak{A}(w)$ — определен$\}$ и (2) если $\mathfrak{A}(w) = \pi$, то $S \underset{\text{lm}}{\overset{\pi}{\Rightarrow}} w$.

Если k-предсказывающий алгоритм анализа $\mathfrak{A}$, использующий управляющую таблицу M, является правильным для cfg G, то говорят, что M — *правильная управляющая таблица* для грамматики G.

Пример 2.4. Построим 1-предсказывающий алгоритм анализа для простой $LL(1)$-грамматики, приведенной в примере 2.2. Пронумеруем ее правила:

1) $S \to aBS$, 2) $S \to b$, 3) $B \to a$, 4) $B \to bSB$.

С учетом свойств этой грамматики, выявленных в примере 2.2, нетрудно построить управляющую таблицу анализатора (табл. 2.1).

Табл. 2.1

$X \in \Gamma$	Аванцепочки		
	a	b	ε
S	aBS, 1	b, 2	error
B	a, 3	bSB, 4	error
A	pop	error	error
B	error	pop	error
\$	error	error	accept

Используя эту таблицу, алгоритм $\mathfrak{A}$ анализировал бы входную цепочку *abbab* следующим образом:

$$(abbab, S\$, \varepsilon) \vdash (abbab, aBS\$, 1) \vdash (bbab, BS\$, 1) \vdash (bbab, bSBS\$, 14) \vdash$$
$$\vdash bab, SBS\$, 14) \vdash (bab, bBS\$, 142) \vdash (ab, BS\$, 142) \vdash$$
$$\vdash (ab, aS\$, 1423) \vdash (b, S\$, 1423) \vdash (b, b\$, 14232) \vdash (\varepsilon, \$, 14232).$$

Итак, $\mathfrak{A}(abbab) = 14232$. Нетрудно проверить, что 14232 — действительно левый анализ *abbab*: достаточно лишь произвести левосторонний вывод с использованием правил в указанной последовательности:

$$S \underset{\text{lm}}{\overset{1}{\Rightarrow}} aBS \underset{\text{lm}}{\overset{4}{\Rightarrow}} abSBS \underset{\text{lm}}{\overset{2}{\Rightarrow}} abbBS \underset{\text{lm}}{\overset{3}{\Rightarrow}} abbaS \underset{\text{lm}}{\overset{2}{\Rightarrow}} abbab.$$

Следовательно, $\mathfrak{A}$ — правильный 1-предсказывающий алгоритм анализа для грамматики *G*.

Пример 2.5. Построим детерминированный магазинный преобразователь, моделирующий анализатор предыдущего примера. Поскольку грамматика была простой, то нетрудно заметить, что за движением типа 1 сразу же следует pop-движение, продвигающее анализатор к следующему символу входной цепочки и стирающее верхний символ магазина, который всегда оказывается равным входному. В отличие от анализатора dpdt будет продвигаться по входной цепочке при каждом движении. Соответственно в магазин он сразу будет писать правую часть правила без первого символа. Кроме того, конец входной цепочки будет маркирован, чтобы контролировать его достижение.

Принимая во внимание сказанное, положим

$P = (\{q_0, q, \text{accept}\}, \{a, b, \$\}, \{S, B, a, b, \$\}, \{1, 2, 3, 4\}, \delta, q_0, \$, \{\text{accept}\})$,

где

1) $\delta(q_0, \varepsilon, \$) = (q, S\$, \varepsilon)$ — воспроизводит начальную конфигурацию $\mathfrak{A}$;

2) $\delta(q, a, S) = (q, BS, 1)$ — воспроизводит $M(S, a) = (aBS, 1)$;

3) $\delta(q, b, S) = (q, \varepsilon, 2)$ — воспроизводит $M(S, b) = (b, 2)$;

4) $\delta(q, a, B) = (q, \varepsilon, 3)$ — воспроизводит $M(B, a) = (a, 3)$;

5) $\delta(q, b, B) = (q, SB, 4)$ — воспроизводит $M(B, b) = (bSB, 4)$;

6) $\delta(q, \$, \$) = (\text{accept}, \varepsilon, \varepsilon)$ — сигнализирует о приеме.

Легко проверить, что $(w\$, \pi) \in \tau_e(P)$ тогда и только тогда, когда $\mathfrak{A}(w) = \pi$.

Посмотрим, как действует этот преобразователь на той же входной цепочке *abbab*:

$(q_0, abbab\$, \$, \varepsilon) \vdash (q, abbab\$, S\$, \varepsilon) \vdash (q, bbab\$, BS\$, 1) \vdash (q, bab\$, SBS\$, 14) \vdash$

$\vdash (q, ab\$, BS\$, 142) \vdash (q, b\$, S\$, 1423) \vdash (q, \$, \$, 14232) \vdash (\text{accept}, \varepsilon, \varepsilon, 14232)$.

Как видим, $(abbab\$, 14232) \in \tau_e(P)$.

§ 2.4. Построение 1-предсказывающего алгоритма анализа по *LL*(1)-грамматике

Алгоритм 2.1: построение *LL*(1)-анализатора.

Вход: $G = (V_N, V_T, P, S)$ — *LL*(1)-грамматика.

Выход: правильный $\mathfrak{A}$ — 1-предсказывающий алгоритм анализа для грамматики *G*.

Метод. Положим $\mathfrak{A} = (\Sigma, \Gamma \cup \{\$\}, \Delta, M, X_0, \$)$, где $\Sigma = V_T$, $\Delta = \{1, 2, \ldots, \#P\}$, $\Gamma = V_N \cup V_T$, $X_0 = S$.

Управляющая таблица M определяется на множестве $(\Gamma \cup \{\$\}) \times (\Sigma \cup \{\varepsilon\})$ следующим образом:

1) $M(A, a) = (\alpha, i)$, если $A \to \alpha$ является i-м правилом во множестве правил P, и $a \in \mathrm{FIRST}_1^G(\alpha)$, $a \neq \varepsilon$. Если $\varepsilon \in \mathrm{FIRST}_1^G(\alpha)$, то $M(A, b) = (\alpha, i)$ для всех $b \in \mathrm{FOLLOW}_1^G(A)$;

2) $M(a, a) = \mathrm{pop}$ для всех $a \in \Sigma$;

3) $M(\$, \varepsilon) = \mathrm{accept}$;

4) $M(X, a) = \mathrm{error}$ для всех $(X, a) \in (\Gamma \cup \{\$\}) \times (\Sigma \cup \{\varepsilon\})$, для которых значения элементов, остались не определенными по пп. 1–3.

Пример 2.6. Посредством алгоритма 2.1 построим $LL(1)$-анализатор для $LL(1)$-грамматики $G = (\{E, E', T, T', F\}, \{a, +, *, (,)\}, P, E)$, где

$P = \{(1)\ E \to TE',\ (2)\ E' \to +TE',\ (3)\ E' \to \varepsilon,\ (4)\ T \to FT',\ (5)\ T' \to *FT',$
$(6)\ T' \to \varepsilon,\ (7)\ F \to (E),\ (8)\ F \to a\}$.

Положим $\mathfrak{A} = (\{a, +, *, (,)\}, \{E, T, F, a, +, *, (,), \$\}, \{1, 2, 3, 4, 5, 6, 7, 8\}, M, E, \$)$, где M дана в виде табл. 2.2. В ней пустые клетки соответствуют значениям error.

Табл. 2.2

Маг. сим-ы	Аванцепочки					
	a	+	*	(	)	ε
E	TE', 1			TE', 1		
E'		$+TE'$, 2			ε, 3	ε, 3
T	FT', 4			FT', 4		
T'		ε, 6	$*FT'$, 5		ε, 6	ε, 6
F	a, 8			(E), 7		
a	pop					
+		pop				
*			pop			
(				pop		
)					pop	
$						accept

1-Предсказывающий алгоритм анализа, использующий эту управляющую таблицу, проанализировал бы входную цепочку $(a+a)$, совершив следующую последовательность движений:

$((a+a), E\$, \varepsilon) \vdash ((a+a), TE'\$, 1) \vdash ((a+a), FT'E'\$, 14) \vdash$
$\vdash ((a+a), (E)T'E'\$, 147) \vdash (a+a), E)T'E'\$, 147) \vdash (a+a), TE')T'E'\$, 1471) \vdash$
$\vdash (a+a), FT'E')T'E'\$, 14714) \vdash (a+a), aT'E')T'E'\$, 147148) \vdash$

$\vdash (+a),\ T'E')\ T'E'\$,\ 147148) \vdash (+a),\ E')\ T'E'\$,\ 1471486) \vdash$

$\vdash (+a),+TE')\ T'E'\$,\ 14714862) \vdash (a),\ TE')\ T'E'\$,\ 14714862) \vdash$

$\vdash (a),\ FT'E')\ T'E'\$,\ 147148624) \vdash (a),\ aT'E')\ T'E'\$,\ 1471486248) \vdash$

$\vdash (\ ),\ T'E')T'E'\$,\ 1471486248) \vdash (\ ),\ E')\ T'E'\$,\ 14714862486) \vdash$

$\vdash (\ ),\ )\ T'E'\$,\ 147148624863) \vdash (\varepsilon, T'E'\$,\ 147148624863) \vdash$

$\vdash (\varepsilon,\ E'\$,\ 1471486248636) \vdash (\varepsilon,\ \$,\ 14714862486363).$

Итак, $\mathfrak{A}((a+a)) = 14714862486363$. Нетрудно проверить, что $E \underset{\text{lm}}{\overset{\pi}{\Rightarrow}} (a+a)$, где $\pi = 14714862486363$.

Теорема 2.4. *Алгоритм* 2.1 *производит правильный* 1*-предсказывающий алгоритм анализа для любой LL*(1)*-грамматики*.

Доказательство. Пусть $G = (V_N, V_T, P, S)$ — $LL(1)$-грамматика, и $\mathfrak{A}$ — 1-предсказывающий алгоритм анализа для грамматики G, построенный согласно алгоритму 2.1. Требуется доказать, что $S \underset{\text{lm}}{\overset{\pi}{\Rightarrow}} x$ тогда и только тогда, когда $(x, S\$, \varepsilon) \overset{*}{\vdash} (\varepsilon, \$, \pi)$.

По индукции докажем вспомогательное утверждение, общий смысл которого состоит в том, что анализатор в своем магазине воспроизводит последовательность открытых частей сентенциальных форм левостороннего вывода входной цепочки, если она выводима в данной грамматике G, причем если π — последовательность номеров правил, участвующих в ее выводе, то π образуется на выходе анализатора.

I. Докажем сначала, что если $S \underset{\text{lm}}{\overset{\pi}{\Rightarrow}} x\alpha$, где $x \in \Sigma^*$— закрытая часть, а $\alpha \in (V_N \cup V_T)^*$ — открытая часть данной сентенциальной формы, то для любой цепочки $y \in \Sigma^*$, такой, что $\text{FIRST}_1(y) \in \text{FIRST}_1^G(\alpha)$, анализатор совершает переход $(xy, S\$, \varepsilon) \overset{*}{\vdash} (y, \alpha\$, \pi)$.

Индукция по $l = |\pi|$.

База. Пусть $l = 1$, т.е. $\pi = i$, где i — номер некоторого правила грамматики.

Пусть $S \underset{\text{lm}}{\overset{i}{\Rightarrow}} x\alpha$ и $y \in \Sigma^*$— такая цепочка, что $\text{FIRST}_1(y) \in \text{FIRST}_1^G(\alpha)$. На единственном шаге этого вывода применялось правило вида $S \to x\alpha$, имеющее номер i. Для всех $a \in \text{FIRST}_1^G(x\alpha\,\text{FOLLOW}_1^G(S))$ согласно алгоритму 2.1 $M(S, a) = (x\alpha, i)$

.

Посмотрим, как будет действовать анализатор, начиная с конфигурации $(xy, S\$, \varepsilon)$. Очевидно, что аванцепочка

$$u = \text{FIRST}_1(xy) \in \text{FIRST}_1^G(x\alpha) = \text{FIRST}_1^G(x\alpha\varepsilon) \subseteq \text{FIRST}_1^G(x\alpha\,\text{FOLLOW}_1^G(S))$$

и, следовательно, $M(S, u) = (x\alpha, i)$. Поэтому $(xy, S\$, \varepsilon) \vdash (xy,\ x\alpha\$,\ i) \overset{*}{\vdash} (y,\ \alpha\$, \varepsilon)$, причем последний переход реализуется посредством pop-движений. База доказана.

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

Индукционный переход. Докажем утверждение для $l = n + 1$. Пусть имеется левосторонний вывод длиной $n + 1$: $S \underset{\text{lm}}{\overset{\pi'}{\Rightarrow}} x'\alpha' = x'A\gamma \underset{\text{lm}}{\overset{i}{\Rightarrow}} x'\beta\gamma = x\alpha$. Здесь $\pi = \pi' i$, $\alpha' = A\gamma$, $\beta\gamma = z\alpha$, где $z \in V_T^*$, $x = x'z$, $A \in V_N$, $\beta, \gamma \in V^*$. На последнем шаге вывода применялось правило $A \to \beta$ — i-е правило из множества правил P. Согласно алгоритму 2.1

$$M(A, a) = (\beta, i) \text{ для всех } a \in \text{FIRST}_1^G(\beta\, \text{FOLLOW}_1^G(A)). \qquad (2.1)$$

Посмотрим, как будет действовать анализатор из своей начальной конфигурации $(xy, S\$, \varepsilon) = (x'zy, S\$, \varepsilon) = (x'y', S\$, \varepsilon)$, где $y' = zy$.

Применим к имеющемуся выводу $S \underset{\text{lm}}{\overset{\pi'}{\Rightarrow}} x'\alpha'$ индукционную гипотезу с цепочкой y', поскольку

$$\text{FIRST}_1(y') = \text{FIRST}_1(zy) \in \text{FIRST}_1^G(z\alpha) =$$
$$= \text{FIRST}_1^G(\beta\gamma) \subseteq \text{FIRST}_1^G(A\gamma) = \text{FIRST}_1^G(\alpha').$$

Как следствие получаем

$$(xy, S\$, \varepsilon) = (x'zy, S\$, \varepsilon) = (x'y', S\$, \varepsilon) \vdash^* (y', \alpha'\$, \pi') = (zy,\ A\gamma\$, \pi').$$

Вычислим аванцепочку от zy:

$$u = \text{FIRST}_1(zy) \in \text{FIRST}_1^G(z\alpha) =$$
$$= \text{FIRST}_1^G(\beta\gamma) \subseteq \text{FIRST}_1^G(\beta\, \text{FOLLOW}_1^G(A)),$$

а для таких аванцепочек, как показывает (2.1), $M(A, u) = (\beta, i)$. Поэтому следующее движение и завершающие pop-движения продолжают процесс таким образом:

$$(zy, A\gamma\$, \pi') \vdash (zy, \beta\gamma\$, \pi' i) = (zy, z\alpha\$, \pi' i) \vdash^* (y, \alpha\$, \pi).$$

Что и требовалось.

II. Докажем теперь, что если анализатор совершает переход вида $(xy, S\$, \varepsilon) \vdash^* (y, \alpha\$, \pi)$ для любой цепочки $y \in \Sigma^*$, такой, что $\text{FIRST}_1(y) \in \text{FIRST}_1^G(\alpha)$, то $S \underset{\text{lm}}{\overset{\pi}{\Rightarrow}} x\alpha$, где x — закрытая часть, а α — открытая часть данной сентенциальной формы.

Индукция по $l = |\pi|$.

База. Пусть $l = 1$, т.е. $\pi = i$.

Имеем $(xy, S\$, \varepsilon) \vdash^* (y, \alpha\$, i)$. Анализатор пишет на выходную ленту номер правила только при движении типа 1, а оно совершается только при наличии нетерминала на вершине магазина. Следовательно, это — первое движение, и только оно пишет номер i на выход. Поэтому фактически имеем

$$(xy, S\$, \varepsilon) \vdash (xy, \beta\$, \varepsilon) \vdash^* (y, \alpha\$, i),$$

причем все остальные движения — это pop-движения. Очевидно, что только при $\beta = x\alpha$ достижима завершающая конфигурация. Первое движение обеспечивается элементом таблицы $M(S, a) = (\beta, i)$, где $u = \mathrm{FIRST}_1(xy)$, а это подразумевает существование правила $S \to \beta = x\alpha$ под номером i. Чтобы первое движение могло произойти, необходимо, чтобы $u \in \mathrm{FIRST}_1^G(x\alpha\ \mathrm{FOLLOW}_1^G(S))$. И это действительно так, поскольку

$$u = \mathrm{FIRST}_1(xy) \in \mathrm{FIRST}_1^G(x\alpha\varepsilon) \subseteq \mathrm{FIRST}_1^G(x\alpha\,\mathrm{FOLLOW}_1^G(S)).$$

А тогда $S \underset{\text{lm}}{\overset{i}{\Longrightarrow}} x\alpha$.

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

Индукционный переход. Докажем утверждение для $l = n + 1$.

Пусть

$$(xy, S\$, \varepsilon) = (x'y', S\$, \varepsilon) \vdash^* (y', \alpha'\$, \pi') = (y', A\gamma\$, \pi') \vdash (y', \beta\gamma\$, \pi'i) \vdash^* (y, \alpha\$, \pi),$$

где $\pi = \pi' i$, $|\pi'| = n$, $xy = x'y'$, $\alpha' = A\gamma$. Завершающие pop-движения показывают, что $y' = zy$, $\beta\gamma = z\alpha$; т.к. $xy = x'y' = x'zy$, то $x = x'z$.

Последнее движение типа 1 было выполнено благодаря элементу $M(A, u') = (\beta, i)$ для $u' \in \mathrm{FIRST}_1(y')$, что предполагает существование правила $A \to \beta$ под номером i, а также выполнение условия $u' \in \mathrm{FIRST}_1^G(\beta\,\mathrm{FOLLOW}_1^G(A))$. Оно действительно выполняется, поскольку

$$u' \in \mathrm{FIRST}_1(y') = \mathrm{FIRST}_1(zy) \in \mathrm{FIRST}_1^G(z\alpha) = \mathrm{FIRST}_1^G(\beta\gamma) \subseteq \mathrm{FIRST}_1^G(\beta\,\mathrm{FOLLOW}_1^G(A)).$$

Одновременно можно воспользоваться индукционной гипотезой в применении к $(x'y', S\$, \varepsilon) \vdash^* (y', \alpha'\$, \pi')$, поскольку

$$\mathrm{FIRST}_1(y') = \mathrm{FIRST}_1(zy) \in \mathrm{FIRST}_1^G(z\alpha) = \mathrm{FIRST}_1^G(\beta\gamma) \subseteq \mathrm{FIRST}_1^G(A\gamma) = \mathrm{FIRST}_1^G(\alpha'),$$

и получить как следствие $S \underset{\text{lm}}{\overset{\pi'}{\Longrightarrow}} x'\alpha' = x'A\gamma \underset{\text{lm}}{\overset{i}{\Longrightarrow}} x'\beta\gamma = x'z\alpha = x\alpha$. Что и требовалось.

Из утверждений I и II при $y = \alpha = \varepsilon$ заключаем, что $S \underset{\text{lm}}{\overset{\pi}{\Longrightarrow}} x$ тогда и только тогда, когда $(x, S\$, \varepsilon) \vdash^* (\varepsilon, \$, \pi)$. А это и означает, что 1-предсказывающий алгоритм анализа, построенный согласно алгоритму 2.1, является правильным для $LL(1)$-грамматики G. Теорема доказана.

Замечание 2.1. Алгоритм 2.1 пригоден для построения анализаторов для $LL(k)$-грамматик и при $k > 1$, если только они сильные. При его применении к сильным $LL(k)$-грамматикам всюду, где используется параметр 1, следует использовать значение k.

Прежде чем перейти к обсуждению *LL*-анализа при $k > 1$, введем в рассмотрение еще одну полезную операцию над языками.

***Определение* 2.10.** Пусть Σ — некоторый алфавит и L_1, L_2 — подмножества Σ^*. Положим

$$L_1 \oplus_k L_2 = \left\{ w \in \Sigma^{*k} \;\middle|\; x \in L_1, y \in L_2, w = \begin{cases} xy, \text{ если } |xy| \le k, \\ \mathrm{FIRST}_k(xy) \text{ в противном случае} \end{cases} \right\}.$$

Пример 2.7. Пусть $L_1 = \{\varepsilon, abb\}$ и $L_2 = \{b, bab\}$. Тогда $L_1 \oplus_2 L_2 = \{b, ba, ab\}$. Операция $\oplus_k$ подобна инфиксной операции FIRST_k.

Лемма 2.1. *Для любой контекстно-свободной грамматики* $G = (V_N, V_T, P, S)$ *и любых цепочек* $\alpha, \beta \in V^*$ *имеет место тождество*

$$\mathrm{FIRST}_k^G(\alpha\beta) = \mathrm{FIRST}_k^G(\alpha) \oplus_k \mathrm{FIRST}_k^G(\beta).$$

Доказательство.

I. Пусть $w \in \mathrm{FIRST}_k^G(\alpha\beta)$. Докажем, что тогда $w \in \mathrm{FIRST}_k^G(\alpha) \oplus_k \mathrm{FIRST}_k^G(\beta)$.

Обозначим $L_1 = \mathrm{FIRST}_k^G(\alpha)$, $L_2 = \mathrm{FIRST}_k^G(\beta)$. Из того, что $w \in \mathrm{FIRST}_k^G(\alpha\beta)$, следует, что $w = \mathrm{FIRST}_k(uv)$, если $\alpha \overset{*}{\Rightarrow} u$, $\beta \overset{*}{\Rightarrow} v$. Последние два вывода означают, что $x = \mathrm{FIRST}_k(u) \in \mathrm{FIRST}_k^G(\alpha) = L_1$.

Аналогично $y = \mathrm{FIRST}_k(v) \in \mathrm{FIRST}_k^G(\beta) = L_2$. Учитывая определение операции $\oplus_k$, заключаем, что $\mathrm{FIRST}_k(xy) \in L_1 \oplus_k L_2$.

Остается убедиться в том, что $\mathrm{FIRST}_k(xy) = \mathrm{FIRST}_k(uv) = w$. Так как $x = \mathrm{FIRST}_k(u)$, а $y = \mathrm{FIRST}_k(v)$, то $u = xu'$ и $v = yv'$ при некоторых $u', v' \in V_T^*$, причем если $|x| < k$, то $u' = \varepsilon$, и если $|y| < k$, то $v' = \varepsilon$. Итак, имеем $uv = xu'yv'$.

Если $|x| = k$, то $w = \mathrm{FIRST}_k(uv) = x = \mathrm{FIRST}_k(xy)$. Если $|x| < k$, то $u' = \varepsilon$ и $w = \mathrm{FIRST}_k(uv) = \mathrm{FIRST}_k(xyv')$, причем, если $|y| < k$, то $v' = \varepsilon$ и тогда $w = \mathrm{FIRST}_k(uv) = \mathrm{FIRST}_k(xy)$, если же $|y| = k$, то $\mathrm{FIRST}_k(uv) = \mathrm{FIRST}_k(xyv') = \mathrm{FIRST}_k(xy)$. Итак,

$$w = \mathrm{FIRST}_k(uv) = \mathrm{FIRST}_k(xy) \in L_1 \oplus_k L_2 = \mathrm{FIRST}_k^G(\alpha) \oplus_k \mathrm{FIRST}_k^G(\beta).$$

II. Пусть $w \in \mathrm{FIRST}_k^G(\alpha) \oplus_k \mathrm{FIRST}_k^G(\beta)$. Докажем, что тогда $w \in \mathrm{FIRST}_k^G(\alpha\beta)$.

Согласно определению операции $\oplus_k$ существуют цепочки $x \in \mathrm{FIRST}_k^G(\alpha)$, $y \in \mathrm{FIRST}_k^G(\beta)$ и $w = \mathrm{FIRST}_k(xy)$. Кроме того, согласно определению FIRST_k^G имеем $\alpha \overset{*}{\Rightarrow} xu$, $\beta \overset{*}{\Rightarrow} yv$ при некоторых $u, v \in V_T^*$ и $\mathrm{FIRST}_k(xuyv) \in \mathrm{FIRST}_k^G(\alpha\beta)$. Остается убедиться в том, что $\mathrm{FIRST}_k(xuyv) = \mathrm{FIRST}_k(xy)$.

Если $|x| = k$, то $\mathrm{FIRST}_k(xuyv) = \mathrm{FIRST}_k(xy) = x$. Если $|x| < k$, то $u = \varepsilon$ и $\mathrm{FIRST}_k(xuyv) = \mathrm{FIRST}_k(xyv)$, причем, если $|y| < k$, то $v = \varepsilon$ и $\mathrm{FIRST}_k(xuyv) = \mathrm{FIRST}_k(xy)$, в противном случае от v ничего не зависит и $\mathrm{FIRST}_k(xuyv) = \mathrm{FIRST}_k(xy)$. Следовательно, $w = \mathrm{FIRST}_k(xy) = \mathrm{FIRST}_k(xuyv) \in \mathrm{FIRST}_k(\alpha\beta)$.

Из рассуждений I и II следует утверждение леммы. Что и требовалось.

§ 2.5. Анализ в *LL*(*k*)-грамматиках

В общем случае анализа в *LL*(*k*)-грамматиках по нетерминалу и аванцепочке невозможно однозначно определить правило для построения очередной сентенциальной формы. Рассмотрим, например, *LL*(2)-грамматику из примера 2.3: $S \to aAaa \mid bAba;\ \ A \to b \mid \varepsilon$.

Согласно алгоритму 2.1 для замены нетерминала A в выводе $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha \underset{lm}{\overset{*}{\Rightarrow}} wx$ следует применять правило $A \to \beta$, если аванцепочка $u = \mathrm{FIRST}_2^G(x)$ принадлежит множеству $\mathrm{FIRST}_2^G(\beta\,\mathrm{FOLLOW}_2^G(A))$. Очевидно, что в нашем примере $\mathrm{FOLLOW}_2^G(A) = \{aa, ba\}$. Получается так, что следует применять правило $A \to b$, если аванцепочка из $\mathrm{FIRST}_2^G(b\,\mathrm{FOLLOW}_2^G(A)) = \mathrm{FIRST}_2^G(b\{aa, ba\}) = \{ba, bb\}$, или правило $A \to \varepsilon$, если аванцепочка из множества $\mathrm{FIRST}_2^G(\varepsilon\,\mathrm{FOLLOW}_2^G(A)) = \{aa, ba\}$. Соображения о том, как определять, какое правило из этих двух использовать для замены нетерминала A, если аванцепочка равна ba, уже были рассмотрены в упомянутом примере. Эту неопределенность можно разрешить, если для определения правила, по которому следует раскрывать нетерминал, помимо нетерминала и аванцепочки использовать еще один параметр: $T_{A,L}$ — так называемую *LL*(*k*)-таблицу[14], ассоциированную с двумя индексами, первый из которых — нетерминал A, второй — множество L, вычисляемое с учетом следующих рассуждений.

Пусть имеется вывод $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$ в некоторой *LL*(*k*)-грамматике. Вспоминая теорему 2.1, нетрудно сообразить, что критерием выбора правила $A \to \beta$ для продолжения этого вывода может служить факт принадлежности текущей аванцепочки множеству

$$\mathrm{FIRST}_k^G(\beta\alpha) = \mathrm{FIRST}_k^G(\beta) \oplus_k \mathrm{FIRST}_k^G(\alpha) = \mathrm{FIRST}_k^G(\beta) \oplus_k L,$$

где $L = \mathrm{FIRST}_k^G(\alpha)$. Это множество L и есть тот второй индекс таблицы $T_{A,L}$, о котором шла речь. По аванцепочке таблица $T_{A,L}$ выдает правило, которое следует использовать для замены нетерминала A в текущей левосентенциальной форме.

[14] Не путать с управляющей таблицей *k*-предсказывающего алгоритма анализа.

Ясно, что для любой cfg число таких $LL(k)$-таблиц конечно и все таблицы, необходимые для анализа в данной грамматике, могут быть построены заблаговременно. Они используются k-предсказывающим алгоритмом анализа в магазинных цепочках взамен нетерминалов (см. алгоритм 2.2 далее).

Теперь дадим точное определение $LL(k)$-таблиц и опишем способ построения множества $LL(k)$-таблиц, необходимых для анализа в данной $LL(k)$-грамматике.

***Определение* 2.11.** Пусть $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика. Для каждого $A \in V_N$ и $L \subseteq V_T^{*k}$ определим $T_{A,L}$ — LL(k)-таблицу, ассоциированную с A и L как функцию, которая по данной аванцепочке $u \in V_T^{*k}$ выдает либо значение error, либо A-правило[15] и конечный список подмножеств V_T^{*k}. Именно:

1) $T_{A,L}(u) =$ error, если не существует ни одного правила вида $A \to \alpha \in P$, такого, что $u \in \mathrm{FIRST}_k^G(\alpha) \oplus_k L$;

2) $T_{A,L}(u) = (A \to \alpha, \langle Y_1, Y_2, \ldots, Y_m \rangle)$, если $A \to \alpha$ — единственное правило из P, такое, что $u \in \mathrm{FIRST}_k^G(\alpha) \oplus_k L$; при этом, если $\alpha = x_0 B_1 x_1 B_2 \ldots B_m x_m$, $B_i \in V_N$, $x_j \in V_T^*$, то $Y_i = \mathrm{FIRST}_k^G(x_i B_{i+1} x_{i+1} \ldots B_m x_m) \oplus_k L$, $i = 1,2,\ldots,m$; $j = 0,1,\ldots,m$; $m \geq 0$;

3) $T_{A,L}(u) =$ undefined, если существует несколько A-правил: $A \to \alpha_1 | \alpha_2 | \ldots | \alpha_n$, таких, что $u \in \mathrm{FIRST}_k^G(\alpha_i) \oplus_k L$ для $1 \leq i \leq n$, $n \geq 2$.

Множество терминальных цепочек Y_i называется *множеством локальных правых контекстов для* B_i. В частности, если $m = 0$, то $T_{A,L}(u) = (A \to \alpha, \varnothing)$. Случай 3 для $LL(k)$-грамматик не актуален.

Пусть мы имеем левосторонний вывод в $LL(k)$-грамматике: $S \underset{\text{lm}}{\overset{*}{\Rightarrow}} wA\gamma \underset{\text{lm}}{\overset{*}{\Rightarrow}} wx$. Как было сказано в начале этого параграфа, именно таблица $T_{A,L}$, где $L = \mathrm{FIRST}_k^G(\gamma)$, сообщит, что следующую за $wA\gamma$ сентенциальную форму следует получать при помощи правила $A \to \alpha$, если для $u = \mathrm{FIRST}_k(x)$ окажется, что $T_{A,L}(u) = (A \to \alpha, \langle Y_1, Y_2, \ldots, Y_m \rangle)$. Если $\alpha = x_0 B_1 x_1 B_2 \ldots B_m x_m$, то следующая за $wA\gamma$ сентенциальная форма будет $w\alpha\gamma = wx_0 B_1 x_1 B_2 \ldots B_m x_m \gamma$, и в свое время раскрытием B_1 будет руководить $LL(k)$-таблица T_{B_1,Y_1}, а раскрытием B_m — $LL(k)$-таблица T_{B_m,Y_m}.

Пример 2.8. Рассмотрим уже не раз обсуждавшуюся $LL(2)$-грамматику с правилами $S \to aAaa \mid bAba$; $A \to b \mid \varepsilon$ и построим все $LL(2)$-таблицы, необходимые для анализа в этой грамматике.

Начнем с таблицы $T_{S,\{\varepsilon\}}$: именно она диктует выбор первого правила для раскрытия нетерминала S. Используя правило $S \to aAaa$, вычисляем $\mathrm{FIRST}_2^G(aAaa) \oplus_2 \{\varepsilon\} = \{ab, aa\}$. Используя правило $S \to bAba$, вычисляем $\mathrm{FIRST}_2^G(bAba) \oplus_2 \{\varepsilon\} = \{bb\}$. Соответственно определяем $LL(2)$-таблицу: $T_0 = T_{S,\{\varepsilon\}}$ (табл. 2.3).

[15] A-Правило — правило cfg с нетерминалом A в левой части.

Табл. 2.3

$LL(2)$-табл.	u	$T(u)$	
		Правило	Мн-во лок. прав. конт-в
$T_0 = T_{S,\{\varepsilon\}}$	aa ab bb	$S \to aAaa$ $S \to aAaa$ $S \to bAba$	$\langle\{\, aa \,\}\rangle$ $\langle\{\, aa \,\}\rangle$ $\langle\{\, ba \,\}\rangle$
$T_1 = T_{A,\{aa\}}$	ba aa	$A \to b$ $A \to \varepsilon$	— —
$T_2 = T_{A,\{ba\}}$	bb ba	$A \to b$ $A \to \varepsilon$	— —

Глядя на T_0, легко определить, что потребуются еще таблицы $T_1 = T_{A,\{aa\}}$ и $T_2 = T_{A,\{ba\}}$. Таблица T_1 получается с учетом того, что для правила $A \to b$ получаем $\mathrm{FIRST}_2^G(b) \oplus_2 \{aa\} = \{ba\}$, а для правила $A \to \varepsilon$ — $\mathrm{FIRST}_2^G(\varepsilon) \oplus_2 \{aa\} = \{aa\}$. Таблица T_2 получается с учетом того, что для правила $A \to b$ имеем $\mathrm{FIRST}_2^G(b) \oplus_2 \{ba\} = \{bb\}$, а для правила $A \to \varepsilon$ — $\mathrm{FIRST}_2^G(\varepsilon) \oplus_2 \{ba\} = \{ba\}$. В правых частях правил для нетерминала A нет ни одного нетерминала. Поэтому в двух последних таблицах множества локальных правых контекстов пусты.

При построении этих $LL(2)$-таблиц мы придерживались простой дисциплины: начинали с построения начальной таблицы $T_{S,\{\varepsilon\}}$, а затем в каждой из уже построенных $LL(2)$-таблиц использовали значения элементов, чтобы определить пары индексов других необходимых таблиц — каждое вхождение нетерминала в правило сочеталось с соответствующим локальным множеством. Этот порядок построения $LL(k)$-таблиц фиксируется в следующем описании алгоритма:

Алгоритм 2.2: построение множества $LL(k)$-таблиц, необходимых для анализа в данной $LL(k)$-грамматике.

Вход: $G = (V_N, V_T, P, S)$ — $LL(k)$-грамматика.

Выход: $\mathcal{T}$ — множество $LL(k)$-таблиц, необходимых для анализа в грамматике G.

Метод.

1. Построить $T_0 = T_{S,\{\varepsilon\}}$ и $\mathcal{T} = \{T_0\}$.

2. Если $T_{A,L} \in \mathcal{T}$ и для некоторой цепочки $u \in V_T^{*k}$ и $T_{A,L}(u) = (A \to x_0 B_1 x_1 B_2 \ldots B_m x_m, \langle Y_1, Y_2, \ldots, Y_m \rangle)$, то к множеству таблиц $\mathcal{T}$ добавить те таблицы из множества $\{T_{B_i,Y_i} \mid i = 1, 2, \ldots, m\}$, которых нет в $\mathcal{T}$.

3. Повторять шаг 2 до тех пор, пока ни одну новую $LL(k)$-таблицу не удастся добавить к $\mathcal{T}$.

Такой момент обязательно настанет, так как для любой данной cfg G существует только конечное число таких таблиц (число нетерминалов — конечно, число подмножеств $L \subseteq V_{\mathrm{T}}^{*k}$ тоже конечно). Фактически же для анализа требуются не все возможные $LL(k)$-таблицы, которые можно построить для грамматики G, а только те, которые определяются описанным алгоритмом.

Алгоритм 2.3: построение k-предсказывающего алгоритма анализа.

Вход: $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, S)$ — $LL(k)$-грамматика.

Выход: $\mathfrak{A}$ — правильный k-предсказывающий алгоритм анализа для G.

Метод.

1. Построим $\mathscr{T}$— множество необходимых $LL(k)$-таблиц для грамматики G.

2. Положим $\mathfrak{A} = (\Sigma, \Gamma \cup \{\$\}, \Delta, M, T_0, \$)$, где $\Sigma = V_{\mathrm{T}}$, $\Delta = \{1, 2, \ldots, \#P\}$, $\Gamma = \mathscr{T} \cup V_{\mathrm{T}}$, где $T_0 = T_{S,\{\varepsilon\}}$.

3. Управляющую таблицу M определим на множестве $(\Gamma \cup \{\$\}) \times \Sigma^{*k}$ следующим образом:

3.1. $M(T_{A,L}, u) = (x_0 T_{B_1,Y_1} x_1 T_{B_2,Y_2} \ldots T_{B_m,Y_m} x_m, i)$, если $T_{A,L}(u) = (A \to x_0 B_1 x_1 B_2 \ldots B_m x_m, \langle Y_1, Y_2, \ldots, Y_m \rangle)$, и $A \to x_0 B_1 x_1 B_2 \ldots B_m x_m$ является i-м правилом в P.

3.2. $M(a, av) =$ pop для всех $a \in \Sigma$, $v \in \Sigma^{*k}$.

3.3. $M(\$, \varepsilon) =$ accept.

3.4. $M(X, u) =$ error для всех $(X, u) \in (\Gamma \cup \{\$\}) \times \Sigma^{*k}$, для которых значения элементов, остались не определенными по пп. 1–3.

Пример 2.9. Рассмотрим еще раз $LL(2)$-грамматику, обсуждавшуюся в примере 2.3, с правилами

1) $S \to aAaa$, 2) $S \to bAba$, 3) $A \to b$, 4) $A \to \varepsilon$.

Используя уже построенные для нее $LL(2)$-таблицы легко собрать управляющую таблицу для этой грамматики — см. табл. 2.4.

Табл. 2.4

Маг. сим-ы	Аванцепочки						
	aa	*ab*	*ba*	*bb*	*a*	*b*	ε
T_0	aT_1aa, 1	aT_1aa, 1		bT_2ba, 2			
T_1	ε, 4		*b*, 3				
T_2			ε, 4	*b*, 3			
a	pop	pop			pop		
b			pop	pop		pop	
\$							accept

Например, на входной цепочке *bba* этот 2-предсказывающий алгоритм анализа проходит следующие конфигурации:

$$(bba, T_0\$, \varepsilon) \vdash (bba, bT_2ba\$, 2) \vdash (ba, T_2ba\$, 1) \vdash (ba, ba\$, 24) \vdash$$
$$\vdash (a, a\$, 24) \vdash (\varepsilon, \$, 24).$$

В то же время, посредством правил 1 и 2, получаем $S \underset{\text{lm}}{\overset{2}{\Rightarrow}} bAba \underset{\text{lm}}{\overset{4}{\Rightarrow}} bba$.

Теорема 2.5. *Алгоритм* 2.3 *производит правильный k-предсказывающий алгоритм анализа для любой LL(k)-грамматики.*

Доказательство аналогично доказательству предыдущей теоремы. Пусть $G = (V_N, V_T, P, S)$ — $LL(k)$-грамматика и $\mathfrak{A}$ — k-предсказывающий алгоритм анализа для грамматики G, построенный посредством алгоритма 2.2.

Требуется доказать, что $S \underset{lm}{\overset{\pi}{\Rightarrow}} x$ тогда и только тогда, когда $(x, T_0\$, \varepsilon) \overset{*}{\vdash} (\varepsilon, \$, \pi)$.

По индукции докажем вспомогательное утверждение, общий смысл которого состоит в том, что анализатор в своем магазине воспроизводит последовательность образов открытых частей сентенциальных форм левостороннего вывода входной цепочки, если она выводима в данной грамматике G, причем если π — последовательность номеров правил, участвующих в ее выводе, то π образуется на выходе анализатора. Связь между открытыми частями сентенциальных форм и их образами в магазине $LL(k)$-анализатора описывается следующим гомоморфизмом:

$$h(x) = \begin{cases} a, \text{ если } X = a, \ a \in V_T, \\ A, \text{ если } X = T_{A,L} \in \mathcal{T}. \end{cases}$$

Область определения h легко расширить до Γ^*, и мы будем в дальнейшем предполагать, что это сделано.

I. Докажем сначала, что если $S \underset{lm}{\overset{\pi}{\Rightarrow}} x\alpha$, где x — закрытая, а α — открытая часть данной сентенциальной формы, то для любой цепочки $y \in \Sigma^*$, такой, что $\mathrm{FIRST}_k(y) \in \mathrm{FIRST}_k^G(\alpha)$, анализатор совершает переход $(xy, T_0\$, \varepsilon) \overset{*}{\vdash} (y, \tilde{\alpha}\$, \pi)$, где $h(\tilde{\alpha}) = \alpha$. Попросту говоря, если в $\tilde{\alpha}$ заменить $LL(k)$-таблицы на нетерминалы, с которыми они ассоциированы, то получится α.

Индукция по $l = |\pi|$.

База. Пусть $l = 1$, т.е. $\pi = i$, где i — номер некоторого правила грамматики.

Пусть $S \underset{lm}{\overset{i}{\Rightarrow}} x\alpha$ и $y \in \Sigma^*$, такая, что $\mathrm{FIRST}_k(y) \in \mathrm{FIRST}_k^G(\alpha)$. На единственном шаге этого вывода применяется правило вида $S \to x\alpha$, имеющее номер i. Согласно алгоритму 2.3 $M(S, u) = (x\tilde{\alpha}, i)$ для всех $u \in \mathrm{FIRST}_k^G(x\alpha) \oplus_k \{\varepsilon\}$. Посмотрим, как будет действовать $LL(k)$-анализатор, начиная с конфигурации $(xy, T_0\$, \varepsilon)$.

Очевидно, что аванцепочка $u = \mathrm{FIRST}_k(xy) \in \mathrm{FIRST}_k^G(x\alpha) = \mathrm{FIRST}_k^G(x\alpha) \oplus_k \{\varepsilon\}$ и, следовательно, $M(T_0, u) = (x\tilde{\alpha}, i)$. Поэтому $(xy, T_0\$, \varepsilon) \vdash (xy, x\tilde{\alpha}\$, i) \overset{*}{\vdash} (y, \tilde{\alpha}\$, \varepsilon)$, причем последний переход происходит посредством pop-движений. База доказана.

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

Индукционный переход. Докажем утверждение для $l = n + 1$. Пусть имеется левосторонний вывод длиной $n + 1$: $S \underset{lm}{\overset{\pi'}{\Rightarrow}} x'\alpha' = x'A\gamma \underset{lm}{\overset{i}{\Rightarrow}} x'\beta\gamma = x\alpha$. Здесь

$\pi = \pi' i$, $\alpha' = A\gamma$, $\beta\gamma = z\alpha$, где $z \in V_T^*$, $x = x'z$, $A \in V_N$, $\beta, \gamma \in V^*$. На последнем шаге вывода применялось i-е правило $A \to \beta$ — из множества P.

Согласно алгоритму 2.3

$$M(T_{A,L}, u) = (\tilde{\beta}, i) \text{ для всех } u \in \mathrm{FIRST}_k^G(\beta) \oplus_k L, \text{ где } L = \mathrm{FIRST}_k^G(\gamma). \qquad (2.2)$$

Посмотрим, как будет действовать анализатор из своей начальной конфигурации $(xy, T_0\$, \varepsilon) = (x'zy, T_0\$, \varepsilon) = (x'y', T_0\$, \varepsilon)$, где $y' = zy$. Применим к имеющемуся выводу $S \underset{lm}{\overset{\pi'}{\Rightarrow}} x'\alpha'$ индукционную гипотезу с цепочкой $y' = zy$, поскольку

$$\mathrm{FIRST}_k(y') = \mathrm{FIRST}_k(zy) \in \mathrm{FIRST}_k^G(z\alpha) = \mathrm{FIRST}_k^G(\beta\gamma) \subseteq$$

$$\subseteq \mathrm{FIRST}_k^G(A\gamma) = \mathrm{FIRST}_k^G(\alpha').$$

Как следствие получаем

$$(xy, T_0\$, \varepsilon) = (x'zy, T_0\$, \varepsilon) = (x'y', T_0\$, \varepsilon) \vdash^* (y', \tilde{\alpha}'\$, \pi') = (zy, T_{A,L}\tilde{\gamma}\$, \pi'). \qquad (2.3)$$

Вычислим аванцепочку

$$u = \mathrm{FIRST}_k(zy) \in \mathrm{FIRST}_k^G(z\alpha) = \mathrm{FIRST}_k^G(\beta\gamma) =$$

$$= \mathrm{FIRST}_k^G(\beta) \oplus_k \mathrm{FIRST}_k^G(\gamma) = \mathrm{FIRST}_k^G(\beta) \oplus_k L,$$

а для таких аванцепочек, как показывает (2.2), $M(T_{A,L}, u) = (\tilde{\beta}, i)$. Поэтому следующее движение и завершающие pop-движения продолжают процесс (2.3) следующим образом:

$$(zy,\ T_{A,L}\tilde{\gamma}\$, \pi') \vdash (zy, \tilde{\beta}\tilde{\gamma}\$, \pi' i) = (zy, z\tilde{\alpha}\$, \pi' i) \vdash^* (y, \tilde{\alpha}\$, \pi).$$

Что и требовалось.

II. Докажем теперь, что если $LL(k)$-анализатор совершает переход вида $(xy, T_0\$, \varepsilon) \vdash^* (y, \tilde{\alpha}\$, \pi)$ для любой цепочки $y \in \Sigma^*$, такой, что $\mathrm{FIRST}_k(y) \in \mathrm{FIRST}_k^G(\alpha)$, где $\alpha = h(\tilde{\alpha})$, то $S \underset{lm}{\overset{\pi}{\Rightarrow}} x\alpha$, где x — закрытая, а α — открытая часть данной сентенциальной формы.

Индукция по $l = |\pi|$.

База. Пусть $l = 1$, т.е. $\pi = i$.

Тогда $(xy,\ T_0\$,\ \varepsilon) \vdash^* (y,\ \tilde{\alpha}\$,\ i)$. Анализатор пишет на выходную ленту номер правила только при движении типа 1, а оно совершается только при наличии $LL(k)$-таблицы на вершине магазина. Следовательно, это — первое движение, и только оно пишет номер i на выход. Поэтому фактически имеем

$$(xy,\ T_0\$,\ \varepsilon) \vdash (xy,\ \tilde{\beta}\$,\ \varepsilon) \vdash^* (y,\ \tilde{\alpha}\$,\ i),$$

причем все остальные движения — это pop-движения. Очевидно, что только при $\tilde{\beta} = x\tilde{\alpha}$ достижима завершающая конфигурация посредством одних только pop-движений. Первое движение обеспечивалось элементом таблицы $M(T_0, u) = (\tilde{\beta}, i)$, где $u = \mathrm{FIRST}_k(xy)$, а это подразумевает существование правила $S \to \beta$, в котором $\beta = x\alpha$, под номером i. Тогда $S \underset{lm}{\overset{i}{\Rightarrow}} x\alpha$.

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

Индукционный переход. Докажем утверждение для $l = n + 1$.

Пусть

$$(xy, T_0\$, \varepsilon) = (x'y', T_0\$, \varepsilon) \vdash^* (y', \tilde{\alpha}'\$, \pi') =$$
$$= (y', T_{A,L}\tilde{\gamma}\$, \pi') \vdash (y', \tilde{\beta}\tilde{\gamma}\$, \pi' i) \vdash^* (y, \tilde{\alpha}\$, \pi),$$

где $\pi = \pi' i$, $|\pi'| = n$. Завершающие pop-движения предполагают, что $y' = zy$, $\beta\gamma = z\alpha$. Кроме того, мы имеем $xy = x'y' = x'zy$, откуда $x = x'z$. Так как $\tilde{\alpha}' = T_{A,L}\tilde{\gamma}$, то последнее движение типа 1 было выполнено благодаря элементу $M(T_{A,L}, u) = (\tilde{\beta}, i)$ для $u \in \mathrm{FIRST}_k(y')$, что предполагает существование правила $A \to \beta$ под номером i.

Воспользовавшись индукционной гипотезой в применении к переходу $(x'y', T_0\$, \varepsilon) \vdash^* (y', \alpha'\$, \pi')$, поскольку

$$\mathrm{FIRST}_k(y') = \mathrm{FIRST}_k(zy) \in \mathrm{FIRST}_k^G(z\alpha) = \mathrm{FIRST}_k^G(\beta\gamma) \subseteq$$
$$\subseteq \mathrm{FIRST}_k^G(A\gamma) = \mathrm{FIRST}_k^G(\alpha'),$$

получаем как следствие $S \underset{\text{lm}}{\overset{\pi'}{\Rightarrow}} x'\alpha' = x'A\gamma \underset{\text{lm}}{\overset{i}{\Rightarrow}} x'\beta\gamma = x'z\alpha = x\alpha$. Здесь α, β, γ — гомоморфные образы $\tilde{\alpha}$, $\tilde{\beta}$, $\tilde{\gamma}$ соответственно. Что и требовалось.

Из рассуждений I и II при $y = \alpha = \varepsilon$ заключаем, что $S \underset{\text{lm}}{\overset{\pi}{\Rightarrow}} x$ тогда и только тогда, когда $(x, T_0\$, \varepsilon) \vdash^* (\varepsilon, \$, \pi)$. А это и означает, что k-предсказывающий алгоритм анализа, построенный согласно алгоритму 2.3, является правильным для $LL(k)$-грамматики G. Что и требовалось доказать.

Теорема 2.6. *Число шагов, выполняемых k-предсказывающим алгоритмом анализа, линейно зависит от длины анализируемой цепочки.*

Доказательство. Поскольку k-предсказывающий алгоритм анализа применим только к $LL(k)$-грамматикам, а они не могут быть леворекурсивными (теорема 2.3), то максимальное число шагов в любом левостороннем выводе вида $A \underset{\text{lm}}{\overset{+}{\Rightarrow}} B\alpha$ меньше, чем некоторая константа c, зависящая от грамматики. Во всяком случае, если бы существовал вывод такого вида, длина которого была бы больше числа нетерминалов грамматики $A \Rightarrow B_1\alpha_1 \Rightarrow B_2\alpha_2 \Rightarrow \ldots \Rightarrow B_m\alpha_m = B\alpha$, то какие-то два нетерминала (B_i и B_j при $i \ne j$) неизбежно оказались бы одинаковыми. Другими словами, мы имели бы вывод $B_i\alpha_i \underset{\text{lm}}{\overset{+}{\Rightarrow}} B_j\alpha_j$ при $B_i = B_j$, что означало бы леворекурсивность грамматики.

Поскольку согласно теореме 2.3 $LL(k)$-анализатор аккуратно воспроизводит в своем магазине открытые части сентенциальных форм левостороннего вывода входной цепочки, то участкам вывода вида $B_1\alpha_1 \Rightarrow B_2\alpha_2 \Rightarrow \ldots \Rightarrow B_m\alpha_m$ соответствуют движения типа 1, следующие непосредственно друг за другом, и их число не превосходит c.

Пусть цепочка $x \in L(G)$, где G — $LL(k)$-грамматика и $n = |x|$. Разбирая цепочку x, k-предсказывающий алгоритм анализа, построенный для грамматики G, совершает ровно n pop-движений и не более c движений типа 1 перед каждым из них. Следовательно, общее число движений — не более чем $n + c \times n$. Что и требовалось доказать.

§ 2.6. Тестирование *LL*(*k*)-грамматик

Пусть имеется контекстно-свободная грамматика G. Алгоритмы построения управляющих таблиц $LL(k)$-анализаторов 2.1 и 2.3 достигают успеха только если G — $LL(k)$-грамматика. Поэтому естественно поинтересоваться, является ли данная грамматика G $LL(k)$-грамматикой для данного значения k. Для ответа на этот вопрос можно воспользоваться теоремой 2.1, а для $k = 1$ и сильных $LL(k)$-грамматик также и теоремой 2.2 при данном значении k. Напомним, что в общем случае cfg $G = (V_N, V_T, P, S)$ не является $LL(k)$-грамматикой тогда и только тогда, когда существуют левосторонний вывод вида $S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha$ и пара различных A-правил $A \to \beta$ и $A \to \gamma \in P$ $(\beta \neq \gamma)$, для которых $(\mathrm{FIRST}_k^G(\beta) \oplus_k L) \cap (\mathrm{FIRST}_k^G(\gamma) \oplus_k L) \neq \varnothing$, где $L = \mathrm{FIRST}_k^G(\alpha)$.

Для описания алгоритма, разрешающего этот вопрос, введем вспомогательную функцию.

***Определение* 2.12.** Пусть $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика и $A \in V_N$. Определим $\sigma(A) = \{L \subseteq V_T^{*k} \mid \exists\, S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha, w \in V_T^*$ и $L = \mathrm{FIRST}_k^G(\alpha)\}$.

Алгоритм 2.4: тестирование $LL(k)$-грамматик.

Вход: $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика.

Выход: да, если G — $LL(k)$-грамматика; нет — в противном случае.

Метод.

1. Для каждого нетерминала A, для которого существуют две или более альтернативы, вычисляется $\sigma(A)$.

2. Пусть $A \to \beta$ и $A \to \gamma \in P$ $(\beta \neq \gamma)$ — два различных A-правила. Для каждого $L \in \sigma(A)$ вычисляется $f(L) = (\mathrm{FIRST}_k^G(\beta) \oplus_k L) \cap (\mathrm{FIRST}_k^G(\gamma) \oplus_k L)$. Если $f(L) \neq \varnothing$, то алгоритм завершается с результатом “нет”. Если $f(L) = \varnothing$ для всех $L \in \sigma(A)$, то шаг 2 повторяется для всех пар A-правил.

3. Повторять шаги 1 и 2 для всех нетерминалов из множества V_N.

4. Завершить алгоритм с результатом “да”. Этот шаг выполняется, если были рассмотрены все нетерминалы и алгоритм не завершился на шаге 2.

Разумеется, это описание можно считать алгоритмом, если мы располагаем алгоритмами для вычисления функций $\mathrm{FIRST}_k^G(\beta)$ для $\beta \in V^*$ и $\sigma(A)$ для $A \in V_N$.

§ 2.7. Вычисление функции $\mathbf{FIRST_k^G(\beta)}$

Алгоритм 2.5: вычисление $\mathrm{FIRST}_k^G(\beta)$ для $\beta \in V^*$.

Вход: $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика и $\beta = X_1 X_2 \ldots X_n$, $X_i \in V$ $(i = 1, 2, \ldots, n; n \geq 0)$.

Выход: $\mathrm{FIRST}_k^G(\beta)$.

Метод. Согласно лемме 2.1 $\mathrm{FIRST}_k^G(\beta) = \mathrm{FIRST}_k^G(X_1) \oplus_k \mathrm{FIRST}_k^G(X_2) \oplus_k \ldots \oplus_k \mathrm{FIRST}_k^G(X_n)$, так что задача сводится к вычислению $\mathrm{FIRST}_k^G(X)$ для $X \in V$.

Если $X \in V_T \cup \{\varepsilon\}$, то вычислять нечего, так как в этом случае $\mathrm{FIRST}_k^G(X) = \{X\}$. Остается найти способ вычисления $\mathrm{FIRST}_k^G(X)$ для $X \in V_N$.

Мы будем использовать метод последовательных приближений и строить последовательности множеств $F_i(X)$ для всех $X \in V_N \cup V_T$, $i = 0, 1, 2, \ldots$.

1. $F_i(a) = \{a\}$ для всех $a \in V_T$ и $i \geq 0$.

2. $F_0(A) = \{x \in V_T^{*k} \mid \exists\, A \to x\alpha \in P$, где $|x| = k$, либо $|x| < k$ и $\alpha = \varepsilon\}$.

3. Предположим, что $F_0(A), F_1(A), \ldots, F_{i-1}(A)$ уже построены для всех $A \in V_N$. Тогда

$$F_i(A) = F_{i-1}(A) \cup \{x \in V_T^{*k} \mid \exists\ A \to Y_1 Y_2 \ldots Y_m \in P,\ x \in F_{i-1}(Y_1) \oplus_k F_{i-1}(Y_2) \oplus_k \ldots \oplus_k F_{i-1}(Y_m)\}.$$

4. Так как $F_{i-1}(A) \subseteq F_i(A) \subseteq V_T^{*k}$ для всех $A \in V_N$ и $i > 0$, то шаг 3 требуется повторять до тех пор, пока при некотором $i = j$ не окажется $F_j(A) = F_{j+1}(A)$ для всех $A \in V_N$. Очевидно, что тогда $F_j(A) = F_{j+1}(A) = F_{j+2}(A) = \ldots$.

5. Остается только положить $\mathrm{FIRST}_k^G(A) = F_j(A)$.

Пример 2.10. Рассмотрим еще раз $LL(1)$-грамматику из примера 2.6:
$G = (\{E, E', T, T', F\}, \{a, +, *, (,)\}, P, E)$, где $P = \{(1)\ E \to TE',\ (2)\ E' \to +TE',$
$(3)\ E' \to \varepsilon,\ (4)\ T \to FT',\ (5)\ T' \to *FT',\ (6)\ T' \to \varepsilon,\ (7)\ F \to (E),\ (8)\ F \to a\}$,
и построим функцию $\mathrm{FIRST}_1^G(A)$ для всех $A \in \{E, E', T, T', F\}$.

Прежде всего согласно шагу 2 алгоритма 2.5 получаем

$F_0(E) = F_0(T) = \varnothing$, $F_0(E') = \{+, \varepsilon\}$, $F_0(T') = \{*, \varepsilon\}$, $F_0(F) = \{(, a\}$.

Далее шаг 3 дает

$F_1(E) = F_0(T) \oplus_1 F_0(E') \cup F_0(E) = \varnothing \oplus_1 \{+, \varepsilon\} \cup \varnothing = \varnothing = F_0(E)$;

$F_1(E') = F_0(+) \oplus_1 F_0(T) \oplus_1 F_0(E') \cup F_0(\varepsilon) \cup F_0(E') =$
$= \{+\} \oplus_1 \varnothing \oplus_1 \{+, \varepsilon\} \cup \{\varepsilon\} \cup \{+, \varepsilon\} = \{+, \varepsilon\} = F_0(E')$;

$F_1(T) = F_0(F) \oplus_1 F_0(T') \cup F_0(T) = \{(, a\} \oplus_1 \{*, \varepsilon\} \cup \varnothing = \{(, a\} \neq F_0(T)$;

$F_1(T') = F_0(*) \oplus_1 F_0(F) \oplus_1 F_0(T') \cup F_0(\varepsilon) \cup F_0(T') =$
$= \{*\} \oplus_1 \{(, a\} \oplus_1 \{*, \varepsilon\} \cup \{\varepsilon\} \cup \{*, \varepsilon\} = \{*, \varepsilon\} = F_0(T')$;

$$F_1(F) = F_0(\,(\,) \oplus_1 F_0(E) \oplus_1 F_0(\,)) \cup F_0(a) \cup F_0(F) =$$
$$= \{(\} \oplus_1 \varnothing \oplus_1 \{)\} \cup \{a\} \cup \{(, a\} = \{(, a\} = F_0(F).$$

Поскольку процесс не сошелся для всех нетерминалов, необходимо повторить шаг 3, что дает

$$F_2(E) = F_1(T) \oplus_1 F_1(E') \cup F_1(E) = \{(, a\} \oplus_1 \{+, \varepsilon\} \cup \varnothing = \{(, a\} \neq F_1(E);$$
$$F_2(E') = \{+, \varepsilon\} = F_1(E');$$
$$F_2(T) = \{(, a\} = F_1(T);$$
$$F_2(T') = \{*, \varepsilon\} = F_1(T');$$
$$F_2(F) = \{(, a\} = F_1(F).$$

Так как $F_2(E) \neq F_1(E)$ придется еще раз проделать шаг 3, что дает, наконец,

$$F_3(E) = \{(, a\} = F_2(E);$$
$$F_3(E') = \{+, \varepsilon\} = F_2(E');$$
$$F_3(T) = \{(, a\} = F_2(T);$$
$$F_3(T') = \{*, \varepsilon\} = F_2(T');$$
$$F_3(F) = \{(, a\} = F_2(F).$$

Таким образом, окончательно получаем:

$$\mathrm{FIRST}_1^G(E) = \mathrm{FIRST}_1^G(T) = \mathrm{FIRST}_1^G(F) = \{(, a\};$$
$$\mathrm{FIRST}_1^G(E') = \{+, \varepsilon\};$$
$$\mathrm{FIRST}_1^G(T') = \{*, \varepsilon\}.$$

Теорема 2.7. *Пусть* $G = (V_N, V_T, P, S)$ *— контекстно-свободная грамматика. Алгоритм* 2.5 *правильно вычисляет функцию* $\mathrm{FIRST}_k^G(\beta)$ *для любой цепочки* $\beta \in V^*$ *и* $k \geq 0$.

Доказательство. Фактически достаточно показать, что $F_j(A) = \mathrm{FIRST}_k^G(A)$, если $F_i(A) = F_j(A)$ при $i > j$ для всех $A \in V_N$. Не нарушая общности рассуждений, будем предполагать, что G — приведенная грамматика.

I. Покажем сначала, что $F_j(A) \subseteq \mathrm{FIRST}_k^G(A)$.

Пусть $x \in F_l(A)$ при $l \leq j$. Индукцией по l покажем, что тогда $x \in \mathrm{FIRST}_k^G(A)$.

База. Пусть $l = 0$ и $x \in F_0(A)$. Тогда согласно шагу 2 алгоритма 2.5 существует правило вида $A \to x\alpha \in P$, где $x \in V_T^{*k}$ и $|x| = k$ либо $|x| < k$ и $\alpha = \varepsilon$. Следовательно, $A \overset{*}{\Rightarrow} xy$, где $|x| = k$ и $y \in V_T^*$ либо $|x| < k$ и $y = \varepsilon$. Согласно определению функции FIRST_k^G заключаем, что $x \in \mathrm{FIRST}_k^G(A)$. База доказана.

Индукционная гипотеза. Предположим, что аналогичное утверждение выполняется для всех $l \leq n$ $(0 \leq n \leq j)$.

Индукционный переход. Докажем, что тогда аналогичное утверждение верно для $l = n + 1$.

Пусть $x \in F_{n+1}(A)$. Согласно алгоритму 2.5 либо $x \in F_n(A)$ и тогда в соответствии с индукционным предположением $x \in \mathrm{FIRST}_k^G(A)$ либо существует правило вида $A \to Y_1 Y_2 \ldots Y_m \in P$ и $x \in F_n(Y_1) \oplus_k F_n(Y_2) \oplus_k \ldots \oplus_k F_n(Y_m)$. Если $Y_i \in V_\mathrm{N}$, то в соответствии с индукционной гипотезой $F_n(Y_i) \subseteq \mathrm{FIRST}_k^G(Y_i)$. Если же $Y_i \in V_\mathrm{T}$, то согласно определению множеств F_n и функции FIRST_k^G заключаем, что $F_n(Y_i) = \{Y_i\} = \mathrm{FIRST}_k^G(Y_i)$.

Итак, в любом случае справедливо $F_n(Y_i) \subseteq \mathrm{FIRST}_k^G(Y_i)$ для $Y_i \in V_\mathrm{N} \cup V_\mathrm{T}$. Тогда

$$x \in F_n(Y_1) \oplus_k F_n(Y_2) \oplus_k \ldots \oplus_k F_n(Y_m) \subseteq$$
$$\subseteq \mathrm{FIRST}_k^G(Y_1) \oplus_k \mathrm{FIRST}_k^G(Y_2) \oplus_k \ldots \oplus_k \mathrm{FIRST}_k^G(Y_m) = \mathrm{FIRST}_k^G(Y_1 Y_2 \ldots Y_m) \subseteq$$
$$\subseteq \mathrm{FIRST}_k^G(A),$$

поскольку $A \to Y_1 Y_2 \ldots Y_m \in P$. Что и требовалось доказать.

II. Покажем теперь, что $\mathrm{FIRST}_k^G(A) \subseteq F_j(A)$.

Пусть $x \in \mathrm{FIRST}_k^G(A)$. По определению этой функции существует вывод $A \overset{*}{\Rightarrow} xy$, где $|x| = k$ и $y \in V_\mathrm{T}^*$ либо $|x| < k$ и $y = \varepsilon$. Индукцией по длине вывода l покажем, что если $x \in \mathrm{FIRST}_k^G(A)$ благодаря существованию такого вывода, то $x \in F_j(A)$.

База. Пусть $l = 1$. Имеем $A \Rightarrow xy$, где $|x| = k$ и $y \in V_\mathrm{T}^*$ либо $|x| < k$ и $y = \varepsilon$. Тогда существует правило $A \to xy \in P$ и согласно шагу 2 алгоритма 2.5 $x \in F_0(A) \subseteq F_j(A)$. База доказана.

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

Индукционный переход. Докажем, что тогда аналогичное утверждение верно для $l = n + 1$.

Пусть $A \Rightarrow Y_1 Y_2 \ldots Y_m \overset{n}{\Rightarrow} y_1 y_2 \ldots y_m$ — вывод длиной $n + 1$, где $y_i \in V_\mathrm{T}^*$, $Y_i \overset{n_i}{\Rightarrow} y_i$, $n_i \le n$, $i = 1, 2, \ldots, n$ и $x = \mathrm{FIRST}_k^G(y_1 y_2 \ldots y_m) \in \mathrm{FIRST}_k^G(A)$. Ясно, что

$$x = \mathrm{FIRST}_k^G(y_1 y_2 \ldots y_m) = \mathrm{FIRST}_k^G(y_1) \oplus_k \mathrm{FIRST}_k^G(y_2) \oplus_k \ldots \oplus_k \mathrm{FIRST}_k^G(y_m) \subseteq$$
$$\subseteq \mathrm{FIRST}_k^G(Y_1) \oplus_k \mathrm{FIRST}_k^G(Y_2) \oplus_k \ldots \oplus_k \mathrm{FIRST}_k^G(Y_m) \subseteq$$
$$\subseteq F_n(Y_1) \oplus_k F_n(Y_2) \oplus_k \ldots \oplus_k F_n(Y_m).$$

Последнее вложение множеств имеет место в соответствии со следствием из индукционной гипотезы, примененной к выводам $Y_i \overset{n_i}{\Rightarrow} y_i$, $n_i \le n$, с учетом того, что при $n_i = 0$ имеем $Y_i = y_i$.

Итак, существует правило $A \to Y_1 Y_2 \ldots Y_m \in P$ и $x \in F_n(Y_1) \oplus_k F_n(Y_2) \oplus_k \ldots \oplus_k F_n(Y_m)$. Согласно шагу 3 алгоритма 2.5 заключаем, что $x \in F_{n+1}(A) \subseteq F_j(A)$. Что и требовалось доказать.

Из рассуждений I и II следует равенство $F_j(A) = \mathrm{FIRST}_k^G(A)$, что равнозначно утверждению теоремы.

§ 2.8. Вычисление функции σ(*A*)

Обратимся теперь к описанию алгоритма вычисления функции σ(*A*).

Алгоритм 2.6: вычисление σ(*A*) для $A \in V_N$.

Вход: $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика.

Выход: σ(*A*) для всех $A \in V_N$.

Метод.

По определению $\sigma(A) = \{L \subseteq V_T^{*k} \mid \exists\, S \underset{lm}{\overset{*}{\Rightarrow}} wA\alpha, w \in V_T^* \text{ и } L = \mathrm{FIRST}_k^G(\alpha)\}$.

Введем в рассмотрение еще одну вспомогательную функцию, определяемую для пары нетерминалов следующим образом:

$$\sigma'(A, B) = \{L \subseteq V_T^{*k} \mid \exists\, A \underset{lm}{\overset{*}{\Rightarrow}} wB\alpha,\ w \in V_T^* \text{ и } L = \mathrm{FIRST}_k^G(\alpha)\}.$$

Очевидно, что $\sigma(A) = \sigma'(S, A)$. Таким образом, достаточно дать алгоритм вычисления функции $\sigma'(A, B)$ для любых $A, B \in V_N$. Мы будем вычислять $\sigma'(A, B)$ методом последовательных приближений, параллельно выстраивая последовательности множеств $\sigma_i'(A, B)$ для всех возможных пар $(A, B) \in V_N \times V_N$ следующим образом:

1. $\sigma_0'(A, B) = \{L \subseteq V_T^{*k} \mid \exists\, A \to \beta B\alpha \in P,\ \beta, \alpha \in V_N^*,\ L = \mathrm{FIRST}_k^G(\alpha)\}$.

2. Пусть $\sigma_0'(A, B), \sigma_1'(A, B), \ldots, \sigma_i'(A, B)$ вычислены для всех пар нетерминалов. Положим

$$\sigma_{i+1}'(A,B) = \sigma_i'(A,B) \cup \{L \subseteq V_T^{*k} \mid \exists\, A \to X_1X_2\ldots X_m \in P, L' \in \sigma_i'(X_p, B),\ X_p \in V_N,\ 1 \le p \le m,\ L = L' \oplus_k \mathrm{FIRST}_k^G(X_{p+1}X_{p+2}\ldots X_m)\}.$$

3. Повторять шаг 2 до тех пор, пока при некотором $i = j$ не окажется, что $\sigma_{j+1}'(A,B) = \sigma_j'(A,B)$ для всех $(A,B) \in V_N \times V_N$. Такое j существует, потому что $\sigma_0'(A,B) \subseteq \sigma_1'(A,B) \subseteq \ldots \subseteq 2^{V_T^{*k}}$.

4. Полагаем $\sigma'(A,B) = \sigma_j'(A,B)$.

5. Наконец, $\sigma(A) = \sigma'(S,A)$. Если окажется, в частности, что $\{\varepsilon\} \notin \sigma'(S, S)$, то $\sigma(S) = \sigma'(S, S) \cup \{\{\varepsilon\}\}$.

Пример 2.11. Пусть $G = (\{S, A\}, \{a, b\}, P, S)$, где $P = \{S \to AS, S \to \varepsilon, A \to aA, A \to b\}$.

Табл. 2.5

σ_i'	$i = 0$	$i = 1$
(S, S)	$\{\{\varepsilon\}\}$	$\{\{\varepsilon\}\}$
(S, A)	$\{\{\varepsilon, a, b\}\}$	$\{\{\varepsilon, a, b\}\}$
(A, S)	$\varnothing$	$\varnothing$
(A, A)	$\{\{\varepsilon\}\}$	$\{\{\varepsilon\}\}$

Вычислим функции $\sigma(S)$ и $\sigma(A)$ при $k = 1$. Сначала получаем $\mathrm{FIRST}_1^G(S) = \{\varepsilon, a, b\}$ и $\mathrm{FIRST}_1^G(A) = \{a, b\}$. Затем построим последовательности $\sigma_i'(X, Y)$, где $(X, Y) \in V_N \times V_N$ (табл. 2.5). За два шага получаем $\sigma(S) = \{\{\varepsilon\}\}$, $\sigma(A) = \{\{\varepsilon, a, b\}\}$.

Поясним построение этих последовательностей.

Шаг 1. Построение $\sigma_0'(S, S)$: имеем правило для $S \to AS$, в котором нетерминал S встречается слева и справа. Вычисляется $\mathrm{FIRST}_1^G(\varepsilon) = \{\varepsilon\}$, поскольку правый контекст для S есть ε. Другого правила для S, в котором справа встречается нетерминал S, не существует. Поэтому $\sigma_0'(S, S) = \{\{\varepsilon\}\}$.

Построение $\sigma_0'(S, A)$: существует только одно правило с нетерминалом S в левой части и нетерминалом A — в правой. Это правило $S \to AS$. Вычисляется FIRST_1^G от правого контекста A, т.е. $\mathrm{FIRST}_1^G(S) = \{\varepsilon, a, b\}$. Соответственно $\sigma_0'(S, A) = \{\{\varepsilon, a, b\}\}$.

Построение $\sigma_0'(A, S)$: не существует ни одного правила с нетерминалом A в левой части и нетерминалом S — в правой. Поэтому $\sigma_0'(A, S) = \varnothing$.

Построение $\sigma_0'(A, A)$: имеется только одно продуктивное правило $A \to aA$, дающее по пустому правому контексту A значение $\sigma_0'(A, A) = \{\{\varepsilon\}\}$.

Шаг 2. Построение $\sigma_1'(S, S)$: имеем правило для $S \to AS$, в котором справа два нетерминала — два источника для пополнения множества $\sigma_0'(S, S)$ новыми членами. Именно: можно вычислить новый член по первому вхождению нетерминала A в правую часть этого правила и по второму вхождению нетерминала S:

$L = L' \oplus_1 \mathrm{FIRST}_1^G(S)$, где $L' \in \sigma_0'(A, S) = \varnothing$.

Ясно, что из этого источника никакого пополнения не получится. Попробуем воспользоваться другим источником для получения нового члена:

$L = L' \oplus_1 \mathrm{FIRST}_1^G(\varepsilon)$, где $L' \in \sigma_0'(S, S) = \{\{\varepsilon\}\}$.

Существует единственное множество $L' = \{\varepsilon\}$, с помощью которого вычисляется новый член

$$L = L' \oplus_1 \mathrm{FIRST}_1^G(\varepsilon) = \{\varepsilon\} \oplus_1 \{\varepsilon\} = \{\varepsilon\},$$

но такое множество уже есть в $\sigma_0'(S, S)$.

На рис. 2.2 представлена схема соответствия параметров, используемых при вычислении множества $\sigma_1'(S, S)$.

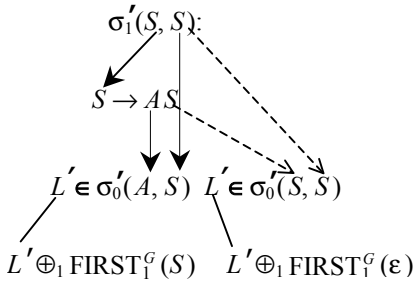

Рис. 2.2.

Аналогичным образом пополняются множества $\sigma_0'(S,A)$, $\sigma_0'(A,S)$ и $\sigma_0'(A,A)$. Однако добавить к этим множествам новые члены, как нетрудно проверить, не удается.

Теперь все готово для тестирования данной cfg G на предмет ее принадлежности к классу $LL(1)$-грамматик. Имеем два альтернативных правила: $S \to AS$, $S \to \varepsilon$ для нетерминала S. Требуется вычислить

$(\mathrm{FIRST}_1^G(AS) \oplus_1 L) \cap (\mathrm{FIRST}_1^G(\varepsilon) \oplus_1 L)$, где $L \in \sigma(S) = \{\{\varepsilon\}\}$.

Получаем

$$(\mathrm{FIRST}_1^G(AS) \oplus_1 L) = \{a, b\} \oplus_1 \{\varepsilon\} = \{a, b\},$$
$$(\mathrm{FIRST}_1^G(\varepsilon) \oplus_1 L) = \{\varepsilon\} \oplus_1 \{\varepsilon\} = \{\varepsilon\}$$

и тогда

$$(\mathrm{FIRST}_1^G(AS) \oplus_1 L) \cap (\mathrm{FIRST}_1^G(\varepsilon) \oplus_1 L) = \{a, b\} \cap \{\varepsilon\} = \varnothing.$$

Имеем также два альтернативных правила $A \to aA$, $A \to b$ для нетерминала A. Требуется вычислить $(\mathrm{FIRST}_1^G(aA) \oplus_1 L) \cap (\mathrm{FIRST}_1^G(b) \oplus_1 L)$, где $L \in \sigma(A) = \{\{\varepsilon, a, b\}\}$. Получаем

$$(\mathrm{FIRST}_1^G(aA) \oplus_1 L) = \{a\} \oplus_1 \{\varepsilon, a, b\} = \{a\},$$
$$(\mathrm{FIRST}_1^G(b) \oplus_1 L) = \{b\} \oplus_1 \{\varepsilon\} = \{b\}$$

и тогда

$$(\mathrm{FIRST}_1^G(aS) \oplus_1 L) \cap (\mathrm{FIRST}_1^G(b) \oplus_1 L) = \{a\} \cap \{b\} = \varnothing.$$

Так как оба пересечения пусты, то данная грамматика G действительно $LL(1)$-грамматика.

Теорема 2.8. Пусть $G = (V_N, V_T, P, S)$ — контекстно-свободная грамматика. Алгоритм 2.6 правильно вычисляет функцию $\sigma(A)$ для любого нетерминала $\sigma \in V_N$ и $k \geq 0$.

Доказательство. Фактически достаточно показать, что $\sigma_j'(A, B) = \sigma'(A, B)$, если $\sigma_i'(A, B) = \sigma_j'(A, B)$ при $i > j$ для всех $A, B \in V_N$. Не нарушая общности рассуждений, будем предполагать, что G — приведенная грамматика.

I. Покажем сначала, что $\sigma_j'(A, B) \subseteq \sigma'(A, B)$.

Индукцией по i покажем, что $\sigma_i'(A, B) \subseteq \sigma'(A, B)$ для любого $i \geq 0$.

База. Пусть $i = 0$, т.е. $L \in \sigma_0'(A, B)$. Согласно шагу 1 алгоритма 2.6 существует правило $A \to \beta B \alpha \in P$, $\beta, \alpha \in V^*$, $L = \mathrm{FIRST}_k^G(\alpha)$. Поскольку грамматика G приведенная, то существует вывод, в частности левосторонний, $A \underset{\text{lm}}{\overset{*}{\Longrightarrow}} wB\alpha$. Следовательно, по определению $L = \mathrm{FIRST}_k^G(\alpha) \in \sigma'(A, B)$. База доказана.

Индукционная гипотеза. Предположим, что аналогичное утверждение выполняется для всех $i \leq n$ $(0 \leq n \leq j)$.

Индукционный переход. Докажем, что тогда аналогичное утверждение верно для $i = n + 1$.

Пусть $L \in \sigma'_{n+1}(A, B)$. Согласно шагу 2 алгоритма 2.6 либо $L \in \sigma'_n(A, B)$ и тогда в соответствии с индукционной гипотезой $L \in \sigma'(A, B)$, либо существуют правило $A \to X_1X_2 \ldots X_m \in P$ и $L' \in \sigma'_n(X_p, B)$, $X_p \in V_N$, $1 \le p \le m$, такие, что $L = L' \oplus_k \mathrm{FIRST}_k^G(X_{p+1}X_{p+2} \ldots X_m)$. Согласно индукционной гипотезе, примененной к $L' \in \sigma'_n(X_p, B)$, можем утверждать, что $L' \in \sigma'(X_p, B)$ и что существует левосторонний вывод вида $X_p \overset{*}{\underset{\text{lm}}{\Rightarrow}} w_pB\gamma$, а $L' = \mathrm{FIRST}_k^G(\gamma)$. Тогда можно построить левосторонний вывод:

$$A \overset{*}{\underset{\text{lm}}{\Rightarrow}} X_1X_2 \ldots X_{p-1}X_pX_{p+1} \ldots X_m \overset{*}{\underset{\text{lm}}{\Rightarrow}} w_1w_2 \ldots w_{p-1}X_pX_{p+1} \ldots X_m \overset{*}{\underset{\text{lm}}{\Rightarrow}}$$
$$\overset{*}{\underset{\text{lm}}{\Rightarrow}} w_1w_2 \ldots w_{p-1}w_pB\gamma X_{p+1} \ldots X_m.$$

Согласно определению $\mathrm{FIRST}_k^G(\gamma X_{p+1} \ldots X_m) \in \sigma'(A, B)$. Кроме того,

$$\mathrm{FIRST}_k^G(\gamma X_{p+1} \ldots X_m) = \mathrm{FIRST}_k^G(\gamma) \oplus_k \mathrm{FIRST}_k^G(X_{p+1} \ldots X_m) =$$
$$= L' \oplus_k \mathrm{FIRST}_k^G(X_{p+1} \ldots X_m) = L.$$

Итак, $L \in \sigma'(A, B)$. Что и требовалось.

II. Покажем теперь, что $\sigma'(A, B) \subseteq \sigma'_j(A, B)$.

Пусть $L \in \sigma'(A,B)$. Это значит, что существует левосторонний вывод $A \overset{l}{\underset{\text{lm}}{\Rightarrow}} wB\alpha$, $w \in V_T^*$ и $L = \mathrm{FIRST}_k^G(\alpha)$. Индукцией по длине вывода l покажем, что $L \in \sigma'_j(A, B)$.

База. Пусть $l = 1$, т.е. $A \underset{\text{lm}}{\Rightarrow} wB\alpha, w \in V_T^*$ и $L = \mathrm{FIRST}_k^G(\alpha)$. Тогда существует правило $A \to wB\alpha$ и согласно шагу 1 алгоритма 2.6 $L = \mathrm{FIRST}_k^G(\alpha) \in \sigma'_0(A, B) \subseteq \sigma'_j(A, B)$. База доказана.

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

Индукционный переход. Докажем, что тогда аналогичное утверждение верно для $l = n + 1$. Пусть

$$A \underset{\text{lm}}{\Rightarrow} X_1X_2 \ldots X_{p-1}X_pX_{p+1} \ldots X_m \overset{n-1}{\underset{\text{lm}}{\Rightarrow}} w_1w_2 \ldots w_{p-1}X_pX_{p+1} \ldots X_m \underset{\text{lm}}{\Rightarrow}$$
$$\underset{\text{lm}}{\Rightarrow} w_1w_2 \ldots w_{p-1}w_pB\gamma X_{p+1} \ldots X_m = wB\alpha$$

— вывод длиной $n + 1$, где $w = w_1w_2 \ldots w_p$, $X_p \to w_pB\gamma \in P$, $\alpha = \gamma X_{p+1} \ldots X_m$, и

$$L = \mathrm{FIRST}_k^G(\alpha) = \mathrm{FIRST}_k^G(\gamma X_{p+1} \ldots X_m) = L' \oplus_k \mathrm{FIRST}_k^G(X_{p+1} \ldots X_m),$$

где $L' = \mathrm{FIRST}_k^G(\gamma)$. Именно благодаря этому обстоятельству $L \in \sigma'(A, B)$.

Одновременно согласно шагу 2 алгоритма 2.6 из существования вывода $X_p \overset{l_p}{\underset{\text{lm}}{\Rightarrow}} w_pB\gamma$, $l_p \le n$, следует по определению, что $L' = \mathrm{FIRST}_k^G(\gamma) \in \sigma'(X_p, B)$, а по ин-

дукционному предположению, что $\sigma'(X_p, B) \subseteq \sigma'_j(X_p, B)$ и, следовательно, $L' \in \sigma'_j(X_p, B)$. Кроме того, имеется правило $A \to X_1X_2\ldots X_pX_{p+1}\ldots X_m \in P$. Согласно шагу 2 алгоритма 2.6 $L = L' \oplus_k \mathrm{FIRST}_k^G(X_{p+1}\ldots X_m)$, где $L' \in \sigma'_j(X_p, B)$ есть элемент $\sigma'_{j+1}(A, B)$. Но $\sigma'_{j+1}(A, B) = \sigma'_j(A, B)$. Итак, $L \in \sigma'_j(A, B)$.

Из рассуждений I и II следует равенство $\sigma'_j(A, B) = \sigma'(A, B)$ и утверждение теоремы.

§ 2.9. Алгоритм вычисления функции $\mathrm{FOLLOW}_k^G(A)$

Сопоставляя определения функций

$$\sigma(A) = \{L \subseteq V_T^{*k} \mid \exists\, S \underset{\text{lm}}{\overset{*}{\Rightarrow}} wA\alpha, w \in V_\mathrm{T}^*,\ L = \mathrm{FIRST}_k^G(\alpha)\}$$

и

$$\mathrm{FOLLOW}_k^G(A) = \{w \in V_\mathrm{T}^* \mid \exists\, S \overset{*}{\underset{G}{\Rightarrow}} \gamma A\alpha \text{ и } w \in \mathrm{FIRST}_k^G(\alpha)\},\ A \in V_\mathrm{N},$$

нетрудно сообразить, что

$$\mathrm{FOLLOW}_k^G(A) = \bigcup_{L \in \sigma(A)} L\ .$$

Это равенство — ключ к модификации алгоритма вычисления $\sigma(\mathrm{A})$, дающей алгоритм вычисления $\mathrm{FOLLOW}_k^G(A)$.

Алгоритм 2.7: вычисление $\mathrm{FOLLOW}_k^G(A)$.

Вход: $G = (V_\mathrm{N}, V_\mathrm{T}, P, S)$ — контекстно-свободная грамматика.

Выход: $\mathrm{FOLLOW}_k^G(A)$ для всех $A \in V_\mathrm{N}$.

Метод.

Определим вспомогательную функцию $\varphi(A, B) = \{w \in \Sigma^{*k} \mid \exists\, A \overset{*}{\underset{G}{\Rightarrow}} \gamma B\alpha,\ w \in \mathrm{FIRST}_k^G(\alpha)\}$, где $A, B \in V_\mathrm{N}$. Очевидно, что $\mathrm{FOLLOW}_k^G(A) = \varphi(S, A)$. Остается определить алгоритм для вычисления функции $\varphi(A, B)$.

Вычисление значения $\varphi(A, B)$ производится по следующим шагам:

1. $\varphi_0(A, B) = \{\, w \in \Sigma^{*k} \mid \exists\, A \to \gamma B\alpha \in P,\ \alpha, \gamma \in V^*,\ w = \mathrm{FIRST}_k^G(\alpha)\}$.

2. Пусть значения $\varphi_0(A, B)$, $\varphi_1(A, B), \ldots,$ $\varphi_i(A, B)$ уже построены для всех $(A, B) \in V_\mathrm{N} \times V_\mathrm{N}$. Тогда

$$\varphi_{i+1}(A, B) = \varphi_i(A, B) \cup \{w \in \Sigma^{*k} \mid \exists\, A \to X_1X_2\ldots X_pX_{p+1}\ldots X_m \in P, X_p \in V_\mathrm{N},\ w \in \varphi_i(X_p, B) \oplus_k \mathrm{FIRST}_k^G(X_{p+1}\ldots X_m)\}.$$

3. Шаг 2 повторяется до тех пор, пока при некотором $i = j$ не окажется $\varphi_{j+1}(A, B) = \varphi_j(A, B)$ для всех $(A, B) \in V_\mathrm{N} \times V_\mathrm{N}$. Такое j существует, ибо $\varphi_0(A, B) \subseteq \varphi_1(A, B) \subseteq \ldots \subseteq \Sigma^{*k}$ при том, что последнее множество является конечным. И тогда $\varphi(A, B) = \varphi_j(A, B)$.

4. Полагаем $\mathrm{FOLLOW}_k^G(A) = \varphi(S, A)$ для всех $A \in V_\mathrm{N} \setminus \{S\}$ и $\mathrm{FOLLOW}_k^G(S) = \varphi(S,S) \cup \{\varepsilon\}$, так как всегда $\varepsilon \in \mathrm{FOLLOW}_k^G(S)$, но может оказаться, что $\varepsilon \notin \varphi(S,S)$.

Пример 2.12. Пусть $G = (\{S, A\}, \{a, b\}, P, S)$, где $P = \{S \to aAaa, S \to bAba, A \to b, A \to \varepsilon\}$.

Построим множества $\mathrm{FOLLOW}_2^G(S)$ и $\mathrm{FOLLOW}_2^G(A)$, используя описанный алгоритм (табл. 2.6).

Табл. 2.6

$N \times N$	φ_0	φ_1
(S, S)	$\varnothing$	$\varnothing$
(S, A)	$\{aa, ba\}$	$\{aa, ba\}$
(A, S)	$\varnothing$	$\varnothing$
(A, A)	$\varnothing$	$\varnothing$

Получаем

$$\mathrm{FOLLOW}_2^G(S) = \varphi_0(S, S) \cup \{\varepsilon\} = \{\varepsilon\};\ \mathrm{FOLLOW}_2^G(A) = \varphi_0(S, A) = \{aa, ba\}.$$

Покажем теперь, что алгоритм 2.7 действительно вычисляет функцию $\mathrm{FOLLOW}_k^G(A)$ для любого нетерминала A.

Теорема 2.9. *Пусть* $G = (V_\mathrm{N}, V_\mathrm{T}, P, S)$ *— контекстно-свободная грамматика. Алгоритм* 2.7 *правильно вычисляет функцию* $\mathrm{FOLLOW}_k^G(A)$ *для любого* $A \in V_\mathrm{N}$ *и* $k \geq 0$.

Доказательство. Фактически достаточно показать, что $\varphi_j(A, B) = \varphi(A,B)$, если $\varphi_i(A, B) = \varphi_j(A, B)$ при $i > j$ для всех $A, B \in V_\mathrm{N}$.

I. Покажем сначала, что $\varphi_j(A, B) \subseteq \varphi(A,B)$ или, что то же самое, если $w \in \varphi_l(A, B)$ при $l \leq j$, то $w \in \varphi(A, B)$, используя индукцию по l.

База. Пусть $l = 0$. Имеем $w \in \varphi_0(A, B)$. Согласно шагу 1 алгоритма 2.7 существует правило $A \to \gamma B \alpha \in P$, $\alpha, \gamma \in V^*$, и $w \in \mathrm{FIRST}_k^G(\alpha)$. Это правило позволяет построить вывод $A \underset{G}{\Rightarrow} \gamma B \alpha$, причем любая цепочка $w \in \mathrm{FIRST}_k^G(\alpha)$ также является элементом множества $\varphi(A,B)$ — таково определение этой функции, т.е. $w \in \varphi(A,B)$. База доказана.

Индукционная гипотеза. Предположим, что аналогичное утверждение выполняется для всех $l \leq n$ $(0 \leq n \leq j)$.

Индукционный переход. Докажем, что тогда аналогичное утверждение верно для $l = n + 1$.

Итак, пусть $w \in \varphi_{n+1}(A, B)$. Согласно шагу 2 алгоритма 2.7 либо $w \in \varphi_n(A, B)$ и тогда согласно индукционной гипотезе $w \in \varphi(A,B)$, либо существует правило $A \to X_1 X_2 \ldots X_p X_{p+1} \ldots X_m \in P$ и $w \in \varphi_n(X_p, B) \oplus_k \mathrm{FIRST}_k^G(X_{p+1} \ldots X_m)$, где $X_p \in V_\mathrm{N}$,

$1 \le p \le m$. Иначе говоря, по определению операции $\oplus_k$ существуют $w_1 \in \varphi_n(X_p, B)$, $w_2 \in \mathrm{FIRST}_k^G(X_{p+1}\ldots X_m)$ и $w = \mathrm{FIRST}_k^G(w_1w_2)$. В соответствии с индукционным предположением из того, что $w_1 \in \varphi_n(X_p, B)$, следует, что $w_1 \in \varphi(X_p, B)$. Это значит, что $X_p \overset{*}{\underset{G}{\Rightarrow}} \gamma B\alpha$ и $w_1 \in \mathrm{FIRST}_k^G(\alpha)$. Кроме того,

$$w = \mathrm{FIRST}_k^G(w_1w_2) = \mathrm{FIRST}_k^G(w_1) \oplus_k \mathrm{FIRST}_k^G(w_2) \subseteq$$

$$\subseteq \mathrm{FIRST}_k^G(\alpha) \oplus_k \mathrm{FIRST}_k^G(X_{p+1}\ldots X_m) = \mathrm{FIRST}_k^G(\alpha X_{p+1}\ldots X_m)$$

и при этом существует вывод $A \underset{G}{\Rightarrow} X_1X_2\ldots X_pX_{p+1}\ldots X_m \overset{*}{\underset{G}{\Rightarrow}} X_1X_2\ldots\gamma B\alpha X_{p+1}\ldots X_m$. Следовательно, $w \in \varphi(A, B)$.

II. Покажем теперь, что $\varphi(A, B) \subseteq \varphi_j(A, B)$. Пусть $w \in \varphi(A, B)$ благодаря тому, что существует вывод $A \overset{l}{\Rightarrow} \gamma B\alpha$ и $w \in \mathrm{FIRST}_k^G(\alpha)$. Индукцией по длине вывода l покажем, что $w \in \varphi_j(A, B)$.

База. Пусть $l = 1$. Имеем $A \underset{G}{\Rightarrow} \gamma B\alpha$ и $w \in \mathrm{FIRST}_k^G(\alpha)$. Тогда существует правило $A \to \gamma B\alpha \in P$ и согласно шагу 1 алгоритма 2.7 $w \in \varphi_0(A, B) \subseteq \varphi_j(A, B)$. База доказана.

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

Индукционный переход. Докажем, что тогда аналогичное утверждение верно для $l = n + 1$. Пусть $w \in \varphi(A, B)$ благодаря тому, что существует вывод длиной $n + 1$ вида

$$A \underset{G}{\Rightarrow} X_1X_2\ldots X_{p-1}X_pX_{p+1}\ldots X_m \overset{*}{\underset{G}{\Rightarrow}} \delta X_p\eta \overset{*}{\underset{G}{\Rightarrow}} \underbrace{\delta\beta}_{\gamma} B \underbrace{\chi\eta}_{\alpha} = \gamma B\alpha,$$

в котором

$$X_1X_2\ldots X_{p-1} \overset{*}{\underset{G}{\Rightarrow}} \delta, \quad X_{p+1}\ldots X_m \overset{*}{\underset{G}{\Rightarrow}} \eta, \quad X_p \overset{*}{\underset{G}{\Rightarrow}} \beta B\chi, \quad \alpha = \chi\eta, \quad \gamma = \delta\beta,$$

$$w \in \mathrm{FIRST}_k^G(\alpha) = \mathrm{FIRST}_k^G(\chi\eta) = \mathrm{FIRST}_k^G(\chi) \oplus_k \mathrm{FIRST}_k^G(\eta) \subseteq$$

$$\subseteq \varphi(X_p, B) \oplus_k \mathrm{FIRST}_k^G(X_{p+1}\ldots X_m) \subseteq \varphi_j(X_p, B) \oplus_k \mathrm{FIRST}_k^G(X_{p+1}\ldots X_m) \subseteq$$

$$\subseteq \varphi_{j+1}(A, B) = \varphi_j(A, B),$$

поскольку $\mathrm{FIRST}_k^G(\chi) \subseteq \varphi(X_p, B) \subseteq \varphi_j(X_p, B)$. Последнее вложение следует в соответствии с индукционной гипотезой из существования вывода $X_p \overset{*}{\underset{G}{\Rightarrow}} \beta B\chi$ длиной не больше n.

Здесь использовано существование правила $A \to X_1X_2\ldots X_pX_{p+1}\ldots X_m \in P$, благодаря которому, учитывая шаг 2 алгоритма 2.7, мы заключили, что $w \in \varphi_{j+1}(A, B)$. Что и требовалось доказать.

Из рассуждений I и II следует равенство $\varphi_j(A, B) = \varphi(A, B)$ и утверждение теоремы.

§ 2.10. *k*-Предсказывающий алгоритм трансляции

Ранее в этой части пособия была доказана теорема 1.3 о том, что выходную цепочку простой семантически однозначной трансляции можно сгенерировать по левостороннему анализу входной цепочки посредством детерминированного магазинного преобразователя. Если входная грамматика схемы синтаксически управляемой трансляции принадлежит классу $LL(k)$, то мы имеем детерминированный механизм, правда, не преобразователь, а k-предсказывающий алгоритм анализа, который выдает левосторонний анализ входной цепочки трансляции, задаваемой такой схемой. Кроме того, в лемме 1.1 этой части был описан способ построения недетерминированного магазинного преобразователя, реализующего трансляцию, определяемую простой схемой. Использованный в ней прием можно применить для модификации k-предсказывающего алгоритма анализа в k-предсказывающий алгоритм трансляции, реализующий трансляцию, специфицируемую простой семантически однозначной sdts с входной грамматикой класса $LL(k)$. Это устройство отличается от $LL(k)$-анализатора лишь тем, что имеет еще один тип движений, состоящий в том, чтобы перенести верхний символ магазина, являющийся дубликатом выходного символа, на выходную ленту, превратив его в настоящий выходной символ (как в лемме 1.1 этой части). Детали прояснятся из описания алгоритма 2.8.

Алгоритм 2.8: построение k-предсказывающего алгоритма трансляции.

Вход: $T = (N, \Sigma, \Delta, R, S)$ — простая семантически однозначная схема синтаксически управляемой трансляции с входной грамматикой G_i класса $LL(k)$.

Выход: $\Im$ — k-предсказывающий алгоритм трансляции, реализующий трансляцию $\tau(T)$.

Метод.

1. Предполагая, что множество $\mathcal{T}$ $LL(k)$-таблиц, необходимых для анализа в грамматике G_i, уже построено, положим $\Im = (\Sigma, \Gamma \cup \{\$\}, \Delta, M, X_0, \$)$, где Σ и Δ — такие же, как в схеме T; $\Gamma = \mathcal{T} \cup \Sigma \cup \Delta'$; $\Delta' = \{b' \mid b' = h(b),\ b \in \Delta\}$; $\Sigma \cap \Delta' = \varnothing$; $X_0 = T_0 = T_{S,\{\varepsilon\}}$ — начальный символ магазина; \$ — маркер "дна" магазина; M: $(\Gamma \cup \{\$\}) \times \Sigma^{*k} \to (\Sigma \cup \Delta')^* \cup \{\text{pop, pass, accept, error}\}$ — управляющая таблица, которая строится по следующим шагам:

2. $M(T_{A,L}, u) = x_0 y_0' T_{A_1,L_1} x_1 y_1' T_{A_2,L_2} x_2 y_2' T_{A_3,L_3} \ldots T_{A_m,L_m} x_m y_m'$, если $T_{A,L}(u) = (A \to x_0 A_1 x_1 A_2 x_2 A_3 \ldots A_m x_m, \langle Y_1, Y_2, Y_3, \ldots, Y_m \rangle)$, и $A \to x_0 A_1 x_1 A_2 x_2 A_3 \ldots A_m x_m, y_0 A_1 y_1 A_2\ y_2 A_3 \ldots A_m y_m \in R$ — i-е правило схемы. Здесь $y_i' = h(y_i)$, $i = 0, 1, 2, \ldots, m$.

3. $M(a, u) = \text{pop}$, если $a \in \Sigma$, $u = av$, $v \in \Sigma^{*k-1}$.

4. $M(b', u) = \text{pass}$ для всех $u \in \Sigma^{*k}$. Такой управляющий элемент определяет переход между конфигурациями: $(x, b'\alpha\$, y) \vdash (x, \alpha\$, yb)$.

5. $M(\$, \varepsilon) = \text{accept}$.

6. $M(X, u) = \text{error}$ для всех $(X, u) \in (\Gamma \cup \{\$\}) \times \Sigma^{*k}$, для которых элементы M не определены по пп. 2–5.

Пример 2.13. Пусть $T = (\{S, A\}, \{a, b\}, \{a, b, e, \langle, \rangle\}, R, S)$, где $R = \{(1)\ S \to aAaa, aAaa; (2)\ S \to bAba, \langle A\rangle a; (3)\ A \to b, b; (4)\ A \to \varepsilon, e\}$.

Входная грамматика этой схемы — не сильная $LL(2)$-грамматика, рассматривавшаяся в предыдущих примерах. В примере 2.9 для нее был построен 2-предсказывающий алгоритм анализа. Применяя алгоритм 2.8 к данной схеме, получаем следующий 2-предсказывающий алгоритм трансляции, реализующий определяемую ею трансляцию:

$\Im = (\{a, b\}, \{T_0, T_1, T_2, a, b, a', b', e, \langle, \rangle, \$\}, \{a, b, e, \langle, \rangle\}, M, T_0, \$)$,

где M представляется управляющей табл. 2.7.

Табл. 2.7

Маг. сим-ы	Аванцепочки						
	aa	ab	ba	bb	a	b	ε
T_0	$aa'T_1aaa'a'$	$aa'T_1aaa'a'$		$b\langle T_2ba\rangle a'$			
T_1	e		bb'				
T_2			e	bb'			
a	pop	pop			pop		
b			pop	pop		pop	
a'	pass	pass	pass	pass	pass	pass	pass
b'	pass	pass	pass	pass	pass	pass	pass
e	pass	pass	pass	pass	pass	pass	pass
$\langle$	pass	pass	pass	pass	pass	pass	pass
$\rangle$	pass	pass	pass	pass	pass	pass	pass
\$							accept

Для иллюстрации работы только что построенного 2-предсказывающего алгоритма трансляции рассмотрим обработку входной цепочки bba:

$(bba, T_0\$, \varepsilon) \vdash (bba, b\langle T_2ba\rangle a'\$, \varepsilon) \vdash (ba, \langle T_2ba\rangle a'\$, \varepsilon) \vdash (ba, T_2ba\rangle a'\$, \langle) \vdash$

$\vdash (ba, eba\rangle a'\$, \langle) \vdash (ba, ba\rangle a'\$, \langle e) \vdash (a, a\rangle a'\$, \langle e) \vdash (\varepsilon, \rangle\$, \langle e) \vdash$

$\vdash (\varepsilon, a'\$, \langle e\rangle) \vdash (\varepsilon, \$, \langle e\rangle a)$.

Итак, $\Im(bba) = <e>a$. Нетрудно проверить, что $(S, S) \overset{*}{\underset{T}{\Rightarrow}} (bba, <e>a)$.

Замечание 2.2. Если входная грамматика схемы — $LL(1)$-грамматика или сильная $LL(k)$-грамматика, то очевидно, что можно обойтись без построения $LL(k)$-таблиц, как показано на следующем примере.

Пример 2.14. Пусть $T = (\{E, E', T, T', F\}, \{a, +, *, (,)\}, \{a, +, *\}, R, E)$, где

$R = \{$ (1) $E \to TE', TE'$; (5) $T' \to *FT', F*T'$;

(2) $E' \to +TE', T+E'$; (6) $T' \to \varepsilon, \varepsilon$;

(3) $E' \to \varepsilon, \varepsilon$; (7) $F \to (E), E$;

(4) $T \to FT', FT'$; (8) $F \to a, a\}$.

Очевидно, что T — простая, семантически однозначная sdts с входной грамматикой $LL(1)$. Посредством алгоритма 2.7 получаем следующий 1-предсказы-вающий алгоритм трансляции:

$$\Im = (\{a, +, *, (,)\}, \{E, E', T, T', F, a, +, *, (,), a', +', *', \$\}, \{a, +, *\}, M, E, \$),$$

где M представлена в табл. 2.8.

Табл. 2.8

Маг. сим-ы	Аванцепочки					
	a	+	*	(	)	ε
E	TE'			TE'		
E'		$+T+'E'$			ε	ε
T	FT'			FT'		
T'		ε	$*F*'T'$		ε	ε
F	aa'			(E)		
a	pop					
+		pop				
*			pop			
(				pop		
)					pop	
a'	pass					
$+'$						
$*'$						
$						accept

Посмотрим, как будет действовать построенный нами 1-предсказывающий алгоритм трансляции на входной цепочке $(a+a)$[16].

$((a+a), E\$, \varepsilon) \vdash ((a+a), TE'\$, \varepsilon) \vdash ((a+a), FT'E'\$, \varepsilon) \vdash$

$\vdash ((a+a), (E)T'E'\$, \varepsilon) \vdash (a+a), E)T'E'\$, \varepsilon) \vdash (a+a), TE')T'E'\$, \varepsilon) \vdash$

$\vdash (a+a), FT'E')T'E'\$, \varepsilon) \vdash (a+a), aa'T'E')T'E'\$, \varepsilon) \vdash$

$\vdash (+a), a'T'E')T'E'\$, \varepsilon) \vdash (+a), T'E')T'E'\$, a) \vdash (+a), E')T'E'\$, a) \vdash$

$\vdash (+a), +T+'E')T'E'\$, a) \vdash (a), T+'E')T'E'\$, a) \vdash$

$\vdash (a), FT'+'E')T'E'\$, a) \vdash (a), aa'T'+'E')T'E'\$, a) \vdash$

$\vdash (), a'T'+'E')T'E'\$, a) \vdash (), T'+'E')T'E'\$, aa) \vdash$

$\vdash (), +'E')T'E'\$, aa) \vdash (), E')T'E'\$, aa+) \vdash (\varepsilon, T'E'\$, aa+) \vdash$

$\vdash (\varepsilon, E'\$, aa+) \vdash (\varepsilon, \$, aa+).$

Итак, $\Im((a+a)) = aa+$. Нетрудно проверить, что $(E, E) \overset{*}{\underset{T}{\Rightarrow}} ((a+a), aa+)$.

[16] Не следует путать скобки, ограничивающие компоненты конфигурации, со скобками — входными символами.

Теорема 2.10. *Пусть* $T = (N, \Sigma, \Delta, R, S)$ *— простая семантически однозначная схема синтаксически управляемой трансляции с входной грамматикой* G_i *класса* $LL(k)$ *и* $\Im = (\Sigma, \Gamma \cup \{\$\}, \Delta, M, T_0, \$)$ *— k-предсказывающий алгоритм трансляции, построенный посредством алгоритма* 2.7, *примененного к данной схеме трансляции T. Тогда* $\tau(T) = \tau(\Im)$.

Доказательство. Справедливость данного утверждения следует из того факта, что $LL(k)$-анализатор, построенный по входной грамматике данной схемы, является правильным. Модификация, превращающая его в $LL(k)$-транслятор, фактически состоит в том, что когда анализатор, моделируя шаг левостороннего вывода, замещает некоторую $LL(k)$-таблицу $T_{A,L}$ образом правой части правила $A \to x_0 A_1 x_1 A_2 \ldots A_m x_m$ вида $x_0 T_{A_1,L_1} x_1 \; T_{A_2,L_2} \ldots T_{A_m,L_m} x_m$, транслятор "подмешивает" к этой магазинной цепочке выходные символы из семантической цепочки соответствующего правила схемы таким образом, что полученная смесь $x_0 y_0' T_{A_1,L_1} x_1 y_1' T_{A_2,L_2} \ldots T_{A_m,L_m} x_m y_m'$ в магазине обеспечивает точное воспроизведение движений анализатора и синхронизированную с ними генерацию соответствующей цепочки на выходной ленте. Действительно, выполнив pop-движения и продвинувшись по фрагменту x_i входной цепочки, который является также и фрагментом правила входной грамматики $A \to x_0 A_1 x_1 A_2 \ldots A_m x_m$, транслятор тут же выдает на выход соответствующий фрагмент y_i выходной цепочки, который также является и фрагментом семантической цепочки соответствующего правила схемы $A \to x_0 A_1 x_1 A_2 \ldots A_m x_m, \; y_0 A_1 y_1 A_2 \ldots A_m y_m$.

Теорема 2.11. *Пусть* $T = (N, \Sigma, \Delta, R, S)$ *— простая семантически однозначная схема синтаксически управляемой трансляции с входной грамматикой* G_i *класса* $LL(k)$. *Существует детерминированный магазинный преобразователь P, такой, что* $\tau_e = \{(x\$, y) \mid (x, y) \in \tau(T)\}$.

Доказательство. Заметим, что входная цепочка трансляции, реализуемой магазинным преобразователем, всегда имеет маркер конца, тогда как схема порождает трансляцию с входами без такого маркера. Маркер необходим, чтобы гарантировать детерминизм магазинного преобразователя.

Доказательство основано на построении dpdt, моделирующего движения k-предсказывающего алгоритма трансляции, адекватного данной схеме, который согласно теореме 2.6 существует.

Итак, пусть построен k-предсказывающий алгоритм трансляции

$$\Im = (\Sigma, \Delta, \Gamma \cup \{\$\}, M, T_0, \$).$$

Положим dpdt

$$P = (Q, \Sigma \cup \{\$\}, \Gamma \cup \{\$\}, \Delta, \delta, q_0, \$, \varnothing),$$

где Σ и Γ — те же, что и в $\Im$; $Q = \{q_0\} \cup \{[u] \mid u \in \Sigma^{*k}\} \cup \{[v\$] \mid v \in \Sigma^{*k-1}\}$.

1. $\delta([\varepsilon], T_0\$, \varepsilon)$ — моделирует начальную конфигурацию $\Im$.

2. $\delta([v], a, T) = ([va], T, \varepsilon)$, $v \in \Sigma^{*k-1}$, $a \in \Sigma$, $T \in \mathcal{T}$ — накопление аванцепочки.

3. $\delta([v], \$, T) = ([v\$], T, \varepsilon)$, $v \in \Sigma^{*k-1}$, $T \in \mathcal{T}$ — накопление короткой аванцепочки.

4. $\delta([u], \varepsilon, T) = ([u], \beta, \varepsilon)$, $u \in \Sigma^{*k}$, $T \in \mathcal{T}$, $M(T, u) = \beta$ — моделирование движения типа 1 .

5. $\delta([v\$], \varepsilon, T) = ([v\$], \beta, \varepsilon)$, $v \in \Sigma^{*k-1}$, $T \in \mathcal{T}$, $M(T, v) = \beta$ — моделирование движения типа 1 при короткой аванцепочке.

6. $\delta([av], \varepsilon, a) = ([v], \varepsilon, \varepsilon)$, $a \in \Sigma$, $v \in \Sigma^{*k-1}$ — моделирование pop-движения.

7. $\delta([av\$], \varepsilon, a) = ([v\$], \varepsilon, \varepsilon)$, $a \in \Sigma$, $v \in \Sigma^{*k-2}$ — моделирование pop-движения при короткой аванцепочке.

8. $\delta([u], \varepsilon, b') = ([u], \varepsilon, b)$, $b \in \Delta$, $v \in \Sigma^{*k}$ — моделирование pass-движения.

9. $\delta([v\$], \varepsilon, b') = ([v\$], \varepsilon, b)$, $b \in \Delta$, $u \in \Sigma^{*k-1}$ — моделирование pass-движения при короткой аванцепочке.

10. $\delta([\$], \varepsilon, \$) = ([\varepsilon], \varepsilon, \varepsilon)$ — моделирование перехода в принимающую конфигурацию.

То, что построенный dpdt P действительно точно моделирует k-предсказывающий алгоритм трансляции $\Im$, нетрудно доказать индукцией по числу движений типа 1 того и другого устройств.

2.11. Непростые *LL*(*k*)-трансляции и магазинные процессоры

Пусть $T = (N, \Sigma, \Delta, R, S)$ — непростая семантически однозначная sdts с входной грамматикой G_i класса $LL(k)$. Для реализации такой трансляции можно ввести еще одну модификацию k-предсказывающего алгоритма анализа, называемую *магазинным процессором*.

Магазинный процессор ведет анализ входной цепочки во входной грамматике и генерирует дерево вывода выходной цепочки в выходной грамматике. Другими словами, он моделирует вывод элемента трансляции, используя магазин для манипуляции над синтаксическими цепочками трансляционных форм, а семантические цепочки этих форм представляет в виде дерева вывода в выходной грамматике, разрастающегося в ходе вывода. В тот момент, когда в процессе анализа он воспроизводит шаг замены крайнего левого нетерминала в синтаксической цепочке текущей трансляционной формы, происходит пристраивание вершин, помеченных символами соответствующей семантической цепочки используемого правила схемы к вершине дерева вывода, представляющей связанное вхождение одноименного нетерминала в семантической цепочке трансляционной формы. Чтобы следить за связями между нетерминалами синтаксической цепочки текущей трансляционной формы с соответствующими вершинами дерева, представляющего семантические цепочки, используются указатели,

хранимые в магазине анализатора при нетерминалах или $LL(k)$-таблицах, ассоциированных с ними. Когда нетерминал (или $LL(k)$-таблица) на вершине магазина подменяется цепочкой, представляющей синтаксический элемент правила схемы, указатель, находящийся при нем (ней), указывает на вершину, к которой надлежит пристроить новые вершины. Указатели же на эти вновь появившиеся вершины помещаются в магазин при связанных вхождениях нетерминалов в синтаксическом элементе правила, о котором шла речь.

Проще всего проиллюстрировать работу магазинного процессора схематически — см. рис. 2.3).

На рис. 2.3, *а* представлена начальная конфигурация магазинного процессора. В магазине кроме маркера "дна" магазина находится только начальная $LL(k)$-таблица T_0 с указателем p_0 на корень строящегося дерева вывода в выходной грамматике результата трансляции входной цепочки. Этот указатель запоминается также в отдельной переменной (Root), чтобы через него получить доступ к дереву после его построения. На рис. 2.3, *б* представлена промежуточная конфигурация магазинного процессора, когда часть дерева вывода построена. Среди его листьев находится вершина, помеченная нетерминалом A. В рассматриваемый момент на вершине магазина находится $LL(k)$-таблица $T_{A,L}$ с указателем p на вершину дерева вывода, к которой будет пристроено поддерево, представляющее семантический элемент правила $A \to \alpha_1 B \beta_1,\ \alpha_2 B \beta_2$, используемого на этом шаге моделирования вывода. В правиле явно выделена пара связанных вхождений одного нетерминала (B). Рис. 2.3, *в* иллюстрирует результат

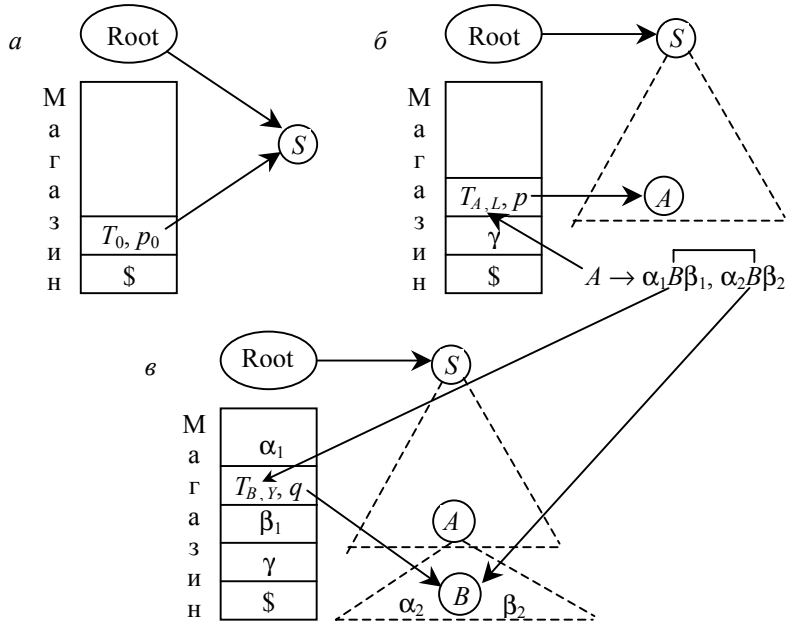

Рис. 2.3.

этого шага: в магазине вместо таблицы $T_{A,L}$ помещен образ синтаксической цепочки $\alpha_1 B \beta_1$, а к узлу A пристроен древовидный образ семантической цепочки $\alpha_2 B \beta_2$. Показана связь одного символа $T_{B,Y}$ — образа нетерминала B в синтаксической цепочке, заместившей в магазине $T_{A,L}$, с узлом B из множества вновь образованных узлов, составляющих образ семантической цепочки $\alpha_2 B \beta_2$. Указатель q определяет ту вершину, к которой будет "подвешиваться" древовидный образ семантической цепочки некоторого правила схемы, когда магазинный символ $T_{B,Y}$ будет замещаться синтаксической цепочкой этого же правила.

Опишем теперь неформально алгоритм построения магазинного процессора по данной схеме, обладающей указанными свойствами.

Алгоритм 2.9: построение магазинного процессора по непростой семантически однозначной sdts с входной грамматикой класса $LL(k)$.

Вход: $T = (N, \Sigma, \Delta, R, S)$ — непростая семантически однозначная sdts с входной грамматикой G_i класса $LL(k)$.

Выход: магазинный процессор $\mathcal{P}$, такой, что $\tau(\mathcal{P}) = \tau(T)$.

Метод.

Магазинный процессор $\mathcal{P}$ в своем магазине будет повторять действия $LL(k)$-анализатора $\mathcal{A}$, построенного по входной грамматике схемы T, используя вместо выходной ленты устройство памяти прямого доступа, в котором он строит дерево вывода результата трансляции в выходной грамматике схемы.

1. Первоначально $\mathcal{P}$ имеет запись (S, p_r)[17] на вершине магазина, где p_r — указатель на корневой узел n_r дерева результата. Этот же указатель дублируются переменной Root.

2. Если $\mathcal{A}$ имеет терминал входной грамматики на вершине магазина и текущий входной символ (первый символ аванцепочки) — такой же терминал, то $\mathcal{A}$ совершает pop-движение и процессор $\mathcal{P}$ делает то же самое.

3. Если $\mathcal{A}$ раскрывает нетерминал A (или замещает $LL(k)$-таблицу, ассоциированную с A) посредством правила $A \to X_1 X_2 \ldots X_m$ с семантическим элементом $y_0 B_1 y_1 \ldots B_m y_m$ и при этом рядом с этим нетерминалом на вершине магазина процессора находится указатель на узел n, то процессор `P` делает следующее:

а) создает узлы прямых потомков для n, помеченных слева направо символами цепочки $y_0 B_1 y_1 \ldots B_m y_m$;

б) на вершине магазина заменяет A (или соответствующую $LL(k)$-таблицу) и указатель при нем на цепочку $X_1 X_2 \ldots X_m$ с указателями при каждом нетерминале, встречающемся в ней; указатель при X_j, если это — нетерминал, указывает на узел, созданный для B_i, если X_j и B_i связаны в правиле схемы $A \to X_1 X_2 \ldots X_m, y_0 B_1 y_1 \ldots B_m y_m$.

---

[17] В случае использования $LL(k)$-таблиц вместо начального нетерминала S будет использоваться начальная $LL(k)$-таблица $T_0 = T_{S,\{\varepsilon\}}$.

4. Если магазин процессора пуст к моменту достижения конца входной цепочки, то он принимает ее. Выход есть дерево с корнем n_r, построенное процессором. Его расположение зафиксировано переменной Root.

Можно показать, что такой алгоритм реализует правильную трансляцию и что на подходящей машине прямого доступа он может быть выполнен за время, линейно зависящее от длины входа.

Теорема 2.12. *Алгоритм* 2.9 *строит магазинный процессор, выдающий в качестве выхода дерево*, *метки листьев которого, выписанные слева направо*, *представляют результат трансляции входной цепочки*.