

# Теория формальных языков и трансляций

2025–2026

## Содержание

<b>I</b>	<b>Языки и их представление</b>	<b>4</b>
1	Алфавиты и языки. Представление языков с помощью распознающих и порождающих процедур. Теорема о рекурсивности языка	4
<b>II</b>	<b>Граматики</b>	<b>4</b>
2	Рекурсивность контекстно-зависимых грамматик	5
3	Деревья вывода в контекстно-свободных грамматиках. Теорема о деревьях вывода	6
<b>III</b>	<b>Конечные автоматы и регулярные грамматики</b>	<b>6</b>
4	Конечные автоматы. Теорема об отношениях эквивалентности и конечных автоматах	6
5	Теорема о единственности конечного автомата с минимальным числом состояний	7
6	Недетерминированные конечные автоматы. Теорема об эквивалентности недетерминированных и детерминированных конечных автоматов	7
7	Конечные автоматы и языки типа 3. Теоремы об эквивалентности конечных автоматов и грамматик типа 3	8
8	Теорема о том, что класс регулярных языков образует булеву алгебру	8
9	Замкнутость регулярных языков относительно произведения и замыкания	8
10	Теорема Клини и следствие из неё	9
<b>IV</b>	<b>Контекстно-свободные грамматики</b>	<b>9</b>
11	Теорема об алгоритмической разрешимости пустоты языка, порождаемого КС-грамматикой	9
12	Теоремы об исключении непродуктивных и недостижимых нетерминалов из КС-грамматик	9
13	Лемма о левостороннем выводе. Теорема об исключении цепных правил из КС-грамматики	9
14	Теорема о нормальной форме Хомского	10
15	Леммы о подстановке и устранении левой рекурсии	10
16	Теорема о нормальной форме Грейбах	10

17 Теорема “ $uvwxy$ ”	10
18 Теоремы об алгоритмической разрешимости конечности КС-языков и исключении нетерминалов, порождающих конечные языки, из КС-грамматик	10
19 Свойство самовставленности. Теорема о регулярности языков, порождаемых несамовставленными КС-грамматиками	11
20 Теорема об $\varepsilon$ -правилах в контекстно-свободных грамматиках	11
21 Специальные типы контекстно-свободных языков и грамматик	11
<b>V Магазинные автоматы</b>	<b>12</b>
22 МП-автомат. Неформальное описание. Формальное определение. Понятие конфигурации	12
23 Теорема об эквивалентности языков, принимаемых недетерминированными магазинными автоматами при конечном состоянии и при пустом магазине	12
24 Эквивалентность недетерминированных магазинных автоматов и контекстно-свободных грамматик	13
<b>VI КСР-грамматики</b>	<b>13</b>
25 Обобщённые регулярные выражения. КС-грамматика в регулярной форме (КСР-грамматика). Примеры	13
26 Определение граф-схемы. Лемма о множестве слов $L(\alpha, \beta)$ . Следствия о множестве слов, порождаемых маршрутами в граф-схеме	13
27 Однозначные регулярные выражения. Утверждения об однозначности регулярных выражений	14
28 Построение графа для нетерминала в КСР-грамматике. Рекуррентный алгоритм построения синтаксической граф-схемы. Лемма 8.2	14
29 Достижимые вершины. Определение множества начальных вершин $H(A)$ и множества конечных вершин $K(A)$	15
30 Понятие достижимости на множестве терминальных вершин. Отношение эквивалентности. Лемма и следствие о языке, порождаемом СГС	15
31 Синтез распознающего автомата для КСР-грамматики. Состояние для регулярного выражения в графе $\Gamma_A$ . Состояние вершины в графе $\Gamma_A$ . Переходное состояние. Регулярный случай	17
32 Синтез распознающего автомата для КСР-грамматики. Состояния в синтаксической граф-схеме. Состояние вершины в СГС. Переходное состояние. Общий случай	18
33 Свойства синтаксической граф-схемы. Леммы о существовании состояний распознавателя в синтаксической граф-схеме	19
34 Регуляризация КС-грамматики. Эквивалентные преобразования. Базисные преобразования. Синтаксическая модель языка	20
35 Алгоритм исключения лево-(право-)рекурсивных нетерминалов в КСР-правиле. Общий случай для всех правил КСР-грамматики	21
36 Схема получения регулярного выражения, эквивалентного приведённой КС-грамматике без самовставлений. Пример	21

<b>VII Трансляции, их представление и реализация</b>	<b>21</b>
1 Некоторые способы задания трансляций: перечисление, гомоморфизм, схемы синтаксически-управляемых трансляций, конечные и магазинные преобразователи	21
2 Простые SDTS. Эквивалентность классов трансляций, задаваемых простыми SDTS и недетерминированными магазинными преобразователями	22
3 Эквивалентность классов трансляций, задаваемых магазинными преобразователями при конечном состоянии и при пустом магазине. Теорема 1.2	23
4 Детерминированная генерация выходной цепочки простой SDT по левостороннему анализу входной цепочки. Теорема 1.3	24
<b>VIII <math>LL(k)</math>-грамматики и трансляции</b>	<b>24</b>
5 Определение и свойства $LL(k)$ -грамматики. Необходимые и достаточные условия принадлежности приведённой КС-грамматики классу $LL(k)$ . Теорема 2.1	24
6 Алгоритм вычисления функции $FIRST_k^G(\beta)$ и его обоснование	25
7 Определение функции $FOLLOW_k^G(\beta)$	25
8 Необходимые и достаточные условия принадлежности приведённой КС-грамматики классу $LL(1)$ . Сильные $LL(k)$ -грамматики. Теорема 2.2	25
9 Достаточные признаки непринадлежности КС-грамматики классу $LL$ . Теорема 2.3	25
10 $k$ -предсказывающие алгоритмы анализа. Формальное определение	26
11 Построение 1-предсказывающего алгоритма анализа по $LL(1)$ -грамматике и его обоснование. Алгоритм 2.1. Теорема 2.4	26
12 Определение операции $\oplus_k$ Лемма 2.1. Обоснование тождества $FIRST_k^G(\alpha\beta) = FIRST_k^G(\alpha) \oplus_k FIRST_k^G(\beta)$	27
13 Анализ в $LL(k)$ -грамматиках. Определение 2.11. $LL(k)$ -таблицы. Алгоритм 2.2: построение множества $LL(k)$ -таблиц, необходимых и достаточных для анализа цепочек в данной $LL(k)$ -грамматике	27
14 Алгоритм 2.3: построение $k$ -предсказывающего алгоритма анализа. Пример 2.9. Теорема 2.5	27
15 Тестирование $LL(k)$ -грамматик. Алгоритм 2.4. Определение 2.12	28
16 Алгоритм 2.5: вычисление функции $FIRST_k^G(\beta)$ и её обоснование. Теорема 2.7	28
17 Вычисление функции $\sigma(A)$ . Алгоритм 2.6. Пример 2.11.	29
18 Алгоритм 2.7 вычисления функции $FOLLOW_k^G(A)$ . Теорема 2.9	30
19 Теорема 2.8 — обоснование правильности вычисления функции $\sigma(A)$ для любого нетерминала $A \in V_N$ и $k \geq 0$	30
20 $k$ -предсказывающий алгоритм трансляции	30
21 Неразрешимые и разрешимые проблемы, касающиеся формальных языков	31
22 Алгоритмически разрешимые проблемы, касающиеся конечных автоматов (проблемы пустоты и бесконечности языков, распознаваемых конечными автоматами, проблема	

# I. Языки и их представление

## 1. Алфавиты и языки. Представление языков с помощью распознающих и порождающих процедур. Теорема о рекурсивности языка

**Определение 1.** *Алфавит* или *словарь* есть конечное множество символов.

**Определение 2.** *Предложение* (*строка*, *слово*) есть любая цепочка конечной длины, составленная из символов некоторого алфавита.

**Определение 3.** Предложение, не содержащее ни одного символа, называется *пустым предложением*.

Обозначение.  $\varepsilon$

**Утверждение 1.** Множество цепочек над алфавитом счётно бесконечно.

**Определение 4.** *Язык* есть любое множество предложений над некоторым алфавитом.

**Определение 5.** *Распознающий алгоритм* определяет, есть ли данное предложение в данном языке. *Распознающая процедура*:

- для предложений в языке прекращает работу с ответом 'да';
- для предложений не из языка завершается с ответом 'нет', или не завершается вовсе.

**Определение 6.** *Порождающая процедура* систематически порождает предложения языка последовательно в некотором порядке.

**Определение 7.** Язык называется *рекурсивно перечислимым*, если существует процедура, которая порождает или распознаёт этот язык.

**Определение 8.** Язык *рекурсивен*, если существует алгоритм его распознавания.

**Теорема 1.**  $L \subseteq V^*$  — язык,  $\bar{L} = V^* \setminus L$   
Если языки  $L$  и  $\bar{L}$  оба рекурсивно перечислимы, то язык  $L$  рекурсивен.

# II. Грамматики

## Формальное определение грамматики. Типы грамматик. Пустое предложение

**Определение 9.** *Грамматикой* называется четвёрка  $G = (V_N, V_T, P, S)$ , где

- $V_N, V_T$  — алфавиты нетерминалов и терминалов соответственно, причём  $V_N \cap V_T = \emptyset$ ;
- $P$  — конечное множество правил, каждое из которых имеет вид  $\alpha \rightarrow \beta$ , где
  - $\alpha \in V^* V_N V^*$ ;
  - $\beta \in V^*$ ;

- $V = V_N \cup V_T$  — объединённый алфавит грамматики;
- $S$  — начальный нетерминал.

**Определение 10.**  $\alpha \rightarrow \beta \in P$  — правило,  $\gamma, \delta$  — цепочки из множества  $V^*$

Будем говорить, что из цепочки  $\gamma\alpha\delta$  непосредственно выводится цепочка  $\gamma\beta\delta$  в грамматике  $G$  при помощи данного правила.

**Обозначение.**  $\gamma\alpha\delta \xRightarrow{G} \gamma\beta\delta$

**Определение 11.**  $\alpha_1, \alpha_2, \dots, \alpha_m$  — цепочки из множества  $V^*$ ,

$$\alpha_1 \xRightarrow{G} \alpha_2, \quad \alpha_2 \xRightarrow{G} \alpha_3, \quad \dots, \quad \alpha_{m-1} \xRightarrow{G} \alpha_m$$

Тогда говорим, что из  $\alpha_1$  выводится  $\alpha_m$  в грамматике  $G$ .

**Обозначение.**  $\alpha_1 \xRightarrow{*G} \alpha_m$

**Определение 12.** Язык, порождаемый грамматикой  $G$  определим как

$$L(G) = \left\{ w \mid w \in V_T^*, \quad S \xRightarrow{*G} w \right\}$$

**Определение 13.** Любая цепочка  $\alpha \in V^*$  такая, что  $S \xRightarrow{*G} \alpha$  называется *сентенциальной формой*.

**Определение 14.** Грамматики  $G_1$  и  $G_2$  называются *эквивалентными*, если  $L(G_1) = L(G_2)$ .

## Типы грамматик

**Определение 15.** Грамматику, на которую не наложено никаких ограничений, назовём *грамматикой типа 0*.

**Определение 16.** Грамматика  $G = (V_N, V_T, P, S)$  является *грамматикой типа 1* или *контекстно-зависимой грамматикой*, если для каждого её правила  $\alpha \rightarrow \beta \in P$  выполняется  $|\beta| \geq |\alpha|$ .

**Определение 17.** Грамматика  $G = (V_N, V_T, P, S)$  является *грамматикой типа 2* или *контекстно-свободной грамматикой*, если каждое её правило имеет вид  $A \rightarrow \beta \in P$ , где  $A \in V_N$ ,  $\beta \in V^+$ .

**Определение 18.** Грамматика  $G = (V_N, V_T, P, S)$  является *грамматикой типа 3* или *регулярной грамматикой*, если каждое её правило имеет вид  $A \rightarrow aB$  или  $A \rightarrow a$ , где  $a \in V_T$ ,  $A, B \in V_N$ .

## Пустое предложение

Расширим данные ранее определения типов грамматик, допустив порождение пустого предложения правилом  $S \rightarrow \varepsilon$ , где  $S$  — начальный символ, при условии, что  $S$  не появляется в правой части никакого правила.

**Лемма 1.**  $G = (V_N, V_T, P, S)$  — грамматика типа  $\geq 1$

Тогда существует другая грамматика  $G_1$  такого же типа, которая порождает тот же самый язык, и в которой ни одно правило не содержит начальный символ в своей правой части.

**Теорема 2.**  $L$  — язык типа  $\geq 1$

Языки  $L \cup \{\varepsilon\}$  и  $L \setminus \{\varepsilon\}$  имеют тот же тип.

## 2. Рекурсивность контекстно-зависимых грамматик

**Определение 19.** Грамматика  $G = (V_N, V_T, P, S)$  *рекурсивна*, если существует алгоритм, который определяет, порождается ли любая данная цепочка  $x \in V_T^*$  грамматикой  $G$ .

**Теорема 3.** Если грамматика контекстно-зависима, то она рекурсивна.

### 3. Деревья вывода в контекстно-свободных грамматиках. Теорема о деревьях вывода

**Определение 20.**  $G = (V_N, V_T, P, S) - \text{cfg}$

Дерево является *деревом вывода* в грамматике  $G$ , если оно удовлетворяет четырём условиям:

1. каждый узел имеет метку — символ из алфавита  $V$ ;
2. метка корня —  $S$ ;
3. если узел имеет по крайней мере одного потомка, то его метка должна быть нетерминалом;
4. если узлы  $n_1, n_2, \dots, n_k$  — прямые потомки узла  $n$ , перечисленные слева-направо, с метками  $A_1, A_2, \dots, A_k$  соответственно, а метка узла  $n$  есть  $A$ , то  $A \rightarrow A_1 A_2 \dots A_k \in P$ .

**Теорема 4.**  $G = (V_N, V_T, P, S) - \text{cfg}$ ,  $\alpha \in V^*$ ,  $\alpha \neq \varepsilon$

Вывод  $S \xRightarrow[G]{*} \alpha$  существует **тогда и только тогда**, когда существует дерево вывода в грамматике  $G$  с результатом  $\alpha$ .

## III. Конечные автоматы и регулярные грамматики

### 4. Конечные автоматы. Теорема об отношениях эквивалентности и конечных автоматах

**Определение 21.** *Конечным автоматом* называется формальная система  $M = (Q, \Sigma, \delta, q_0, F)$ , где

- $Q$  — конечное непустое множество состояний;
- $\Sigma$  — конечный входной алфавит;
- $\delta$  — отображение  $Q \times \Sigma \rightarrow Q$ ;
- $q_0 \in Q$  — начальное состояние;
- $F \subset Q$  — множество конечных состояний.

**Замечание.** Область определения отображения  $\delta$  можно расширить до  $Q \times \Sigma^*$  следующим образом:

$$\delta'(q, \varepsilon) = q, \quad \delta'(q, xa) = \delta(\delta'(q, x), a)$$

**Определение 22.** Цепочка  $x \in \Sigma^*$  *принимается* конечным автоматом  $M$ , если  $\delta(q_0, x) = p$  для некоторого  $p \in F$ .

Множество всех цепочек  $x \in \Sigma^*$ , принимаемых конечным автоматом  $M$ , называется языком, *распознаваемым конечным автоматом*  $M$ , и обозначается  $T(M)$ .

Любое множество цепочек, принимаемых конечным автоматом, называется *регулярным*.

**Определение 23.**  $M - \text{dfa}$

Определим отношение эквивалентности  $R$  на множестве  $\Sigma^*$ :

$$(x, y) \in R \iff \delta(q_0, x) = \delta(q_0, y)$$

**Определение 24.** Отношение эквивалентности называется *право-инвариантным*, если

$$xRy \implies xzRyz \quad \forall z \in \Sigma$$

**Теорема 5.** Следующие утверждения эквивалентны:

1. Язык  $L \subset \Sigma^*$  распознаётся некоторым ф.а.
2. Язык  $L$  есть объединение некоторых классов эквивалентности право-инвариантного отношения эквивалентности конечного индекса.
3. Определим отношение  $R$ :

$$xRy \iff \forall z \in \Sigma^* \quad (xz \in L \iff yz \in L)$$

Отношение  $R$  имеет конечный индекс.

## 5. Теорема о единственности конечного автомата с минимальным числом состояний

**Теорема 6.** Конечный автомат с минимальным числом состояний, распознающий язык  $L$ , единственен с точностью до изоморфизма (переименования состояний), и есть ф.а, задаваемый отношением  $R$  из п. 3 теор. 5.

## 6. Недетерминированные конечные автоматы. Теорема об эквивалентности недетерминированных и детерминированных конечных автоматов

**Определение 25.** Недетерминированным конечным автоматом называется формальная система  $M = (Q, \Sigma, \delta, q_0, F)$ , где

- $Q$  — конечное непустое множество состояний;
- $\Sigma$  — входной алфавит;
- $\delta$  — отображение  $Q \times \Sigma \rightarrow 2^Q$ ;
- $q_0 \in Q$  — начальное состояние;
- $F \subset Q$  — множество конечных состояний.

Область определения  $\delta$  может быть расширена на  $Q \times \Sigma^*$  следующим образом:

$$\delta(q, \varepsilon) = \{q\}, \quad \delta(q, xa) = \bigcup_{p \in \delta(q, x)} \delta(p, a)$$

Область определения  $\delta$  может быть расширена до  $2^Q \times \Sigma^*$  следующим образом:

$$\delta(\{p_1, p_2, \dots, p_k\}, x) = \bigcup_{i=1}^k \delta(p_i, x)$$

**Определение 26.** Цепочка  $x \in \Sigma^*$  принимается НКА  $M$ , если существует состояние  $p \in F$  такое, что  $p \in \delta(q_0, x)$ .

**Обозначение.**  $T(M)$  — множество всех цепочек, принимаемых НКА  $M$ .

**Теорема 7.**  $L$  — язык, распознаваемый НКА.

Тогда существует ДКА, который распознаёт  $L$ .

## 7. Конечные автоматы и языки типа 3. Теоремы об эквивалентности конечных автоматов и грамматик типа 3

**Определение 27.**  $M = (Q, \Sigma, \delta, q_0, F)$  — КА

Конфигурацией конечного автомата  $M$  назовём состояние управления  $q \in Q$  в паре с неп прочитанной частью входной цепочки.

**Теорема 8.**  $G$  — грамматика типа 3.

Тогда существует КА  $M$  такой, что  $T(M) = L(G)$ .

**Теорема 9.**  $M$  — КА.

Существует грамматика  $G$  типа 3 такая, что  $L(G) = T(M)$ .

## 8. Теорема о том, что класс регулярных языков образует булеву алгебру

**Определение 28.** Булева алгебра множеств есть совокупность множеств, замкнутая относительно операций объединения, дополнения и пересечения.

**Определение 29.**  $L \subset \Sigma_1^*$ ,  $\Sigma_1 \subset \Sigma_2$

Под дополнением языка  $L$  подразумевается множество  $L = \Sigma_2^* \setminus L$ .

**Лемма 2.** Класс языков типа 3 замкнут относительно объединения.

**Лемма 3.** Класс множеств, распознаваемых конечными автоматами замкнут относительно дополнения.

**Теорема 10.** Класс множеств, принимаемых конечными автоматами, образует булеву алгебру.

**Теорема 11.** Все конечные множества распознаются конечными автоматами.

## 9. Замкнутость регулярных языков относительно произведения и замыкания

**Определение 30.** Произведением или конкатенацией языков  $L_1$  и  $L_2$  называется множество

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

**Теорема 12.** Класс множеств, распознаваемых конечными автоматами, замкнут относительно произведения.

**Определение 31.** Замыкание языка  $L$  есть множество

$$L^* = \bigcup_{k=0}^{\infty} L^k$$



**Теорема 13.** Класс множеств, принимаемых конечными автоматами, замкнут относительно замыкания.

## 10. Теорема Клини и следствие из неё

**Теорема 14.** Класс множеств, принимаемых конечными автоматами, является наименьшим классом, содержащим все конечные множества, замкнутым относительно объединения, произведения и замыкания.

**Следствие.** Любое выражение, построенное из конечных подмножеств множества  $\Sigma^*$ , где  $\Sigma$  — конечный алфавит, и конечного числа операций объединения, произведения и замыкания со скобками обозначает множество, принимаемое некоторым конечным автоматом.

## IV. Контекстно-свободные грамматики

### 11. Теорема об алгоритмической разрешимости пустоты языка, порождаемого КС-грамматикой

**Теорема 15.** Существует алгоритм для определения, является ли язык, порождаемый данной КС-грамматикой, пустым.

### 12. Теоремы об исключении непродуктивных и недостижимых нетерминалов из КС-грамматик

**Теорема 16.** Для любой КС-грамматики  $G$ , порождающей непустой язык, можно найти эквивалентную КС-грамматику  $G_1 = (V_N^1, V_T, P^1, S)$ , в которой для любого нетерминала  $A$  существует терминальная цепочка  $x$  такая, что  $A \xRightarrow[G]{*} x$ .

**Определение 32.** Нетерминалы из  $V_N^1$  принято называть *продуктивными*.

**Определение 33.** Нетерминалы, которые не участвуют ни в каком выводе сентенциальной цепочки, называются *недостижимыми*.

**Теорема 17.** Для любой КС-грамматики, порождающей непустой язык  $L$ , можно найти КС-грамматику  $G = (V_N, V_T, P, S)$ , порождающую язык  $L$  такую, что для каждого её нетерминала  $A$  существует вывод вида

$$S \xRightarrow[G]{*} x_1 A x_3 \xRightarrow[G]{*} x_1 x_2 x_3, \quad x_1, x_2, x_3 \in V_T^*$$

**Определение 34.** Контекстно-свободные грамматики, удовлетворяющие условию теор. 17, принято называть *приведёнными*.

### 13. Лемма о левостороннем выводе. Теорема об исключении цепных правил из КС-грамматики

**Определение 35.** Вывод в КС-грамматике назовём *левосторонним*, если на каждом его шаге производится замена крайнего левого вхождения нетерминала.

**Обозначение.**  $\xRightarrow[\text{lm}]{}^*$ ,  $\xRightarrow[\text{lm}]{}^*$

**Лемма 4.**  $G = (V_N, V_T, P, S)$  — контекстно-свободная грамматика.

Если  $S \xRightarrow{*}_G x$ , где  $x \in V_T^+$ , то существует и левосторонний вывод  $S \xRightarrow{*}_{lm} x$  в той же грамматике.

**Теорема 18.** Любой контекстно-свободный язык может быть порождён контекстно-свободной грамматикой, не содержащей цепных правил, т. е. правил вида  $A \rightarrow B$ , где  $A$  и  $B$  — нетерминалы.

## 14. Теорема о нормальной форме Хомского

**Теорема 19.** Любой КС-язык может быть порождён грамматикой, в которой все правила имеют вид  $A \rightarrow BC$  или  $A \rightarrow a$  ( $A, B, C$  — нетерминалы,  $a$  — терминал).

## 15. Леммы о подстановке и устранении левой рекурсии

**Лемма 5 (о подстановке).**  $G = (V_N, V_T, P, S)$  — cfg,  $A \rightarrow \alpha_1 B \alpha_2$ ,  $A, B \in V_N$ ,  $\alpha_1, \alpha_2 \in V^*$ ,  $\{B \rightarrow \beta_i \mid \beta_i \in V^+, i = 1, \dots, m\}$  — множество всех  $B$ -порождений, т. е. правил с нетерминалом  $B$  в левой части.

Грамматика  $G_1 = (V_N, V_T, P_1, S)$  получается из грамматики  $G$  отбрасыванием правила  $A \rightarrow \alpha_1 B \alpha_2$  и добавлением правил вида  $A \rightarrow \alpha_1 \beta_i \alpha_2$ .

Тогда  $L(G) = L(G_1)$ .

**Лемма 6 (об устранении левой рекурсии).**  $G = (V_N, V_T, P, S)$  — КС-грамматика,  $\{A \rightarrow A \alpha_i \mid A \in V_N, \alpha_i \in V^*, i = 1, \dots, m\}$  — множество всех леворекурсивных  $A$ -порождений,  $\{A \rightarrow \beta_j \mid j = 1, \dots, n\}$  — множество всех прочих  $A$ -порождений.

$G_1 = (V_N \cup \{Z\}, V_T, P_1, S)$  — КС-грамматика, образованная добавлением нового нетерминала  $Z$  и заменой всех  $A$ -порождений правилами:

- $A \rightarrow \beta_j, A \rightarrow \beta_j Z, j = 1, \dots, n;$
- $Z \rightarrow \alpha_i, Z \rightarrow \alpha_i Z, i = 1, \dots, m.$

Тогда  $L(G_1) = L(G)$ .

## 16. Теорема о нормальной форме Грейбах

**Определение 36.** Говорят, что КС-грамматика  $G = (V_N, V_T, P, S)$  представлена в *нормальной форме Грейбах*, если каждое её правило имеет вид  $A \rightarrow a\alpha$ , где  $a \in V_T, \alpha \in V_N^*$ .

**Теорема 20.** Каждый КС-язык может быть порождён КС-грамматикой в нормальной форме Грейбах.

## 17. Теорема “uvwxy”

**Теорема 21.**  $L$  — cfl

Существуют постоянные  $p$  и  $q$ , зависящие только от языка  $L$ , такие, что если существует  $z \in L$  при  $|z| > p$ , то цепочка  $z$  представима в виде  $z = uvwxu$ , где  $|vwx| \leq q$ , причём  $v$  и  $x$  одновременно не пусты, так что для любого целого  $i \geq 0$  цепочка  $uv^iwx^iy \in L$ .

## 18. Теоремы об алгоритмической разрешимости конечности КС-языков и исключении нетерминалов, порождающих конечные языки, из КС-грамматик

**Теорема 22.** Существует алгоритм для определения, порождает ли данная КС-грамматика  $G$  конечный или бесконечный язык.

**Теорема 23.** Для всякой КС-грамматики  $G_1$  можно найти эквивалентную ей КС-грамматику  $G_2$  такую, что если  $A$  — нетерминал грамматики  $G_2$ , не являющийся начальным нетерминалом, то из  $A$  выводимо бесконечно много терминальных цепочек.

## 19. Свойство самовставленности. Теорема о регулярности языков, порождаемых несамовставленными КС-грамматиками

**Определение 37.** Говорят, что грамматика  $G$  является *самовставленной*, если существует нетерминал  $A$  такой, что  $A \xrightarrow{+}_G \alpha_1 A \alpha_2$ , где  $\alpha_1, \alpha_2 \in V^+$ .

Говорят также, что нетерминал  $A$  является *самовставленным*.

**Теорема 24.**  $G$  — несамовставленная грамматика.

Тогда  $L(G)$  — регулярное множество.

## 20. Теорема об $\varepsilon$ -правилах в контекстно-свободных грамматиках

**Теорема 25.**  $L$  — язык, порождаемый грамматикой  $G = (V_N, V_T, P, S)$ , где каждое правило в  $P$  имеет вид  $A \rightarrow \alpha$ , где  $A$  — нетерминал,  $\alpha \in V^*$ .

Тогда  $L$  может быть порождён грамматикой, в которой каждое правило имеет вид  $A \rightarrow \alpha$ , где  $A$  — нетерминал,  $\alpha \in V^+$ , либо  $S \rightarrow \varepsilon$ , и кроме того, начальный терминал грамматики  $S$  не появляется в правой части никакого правила.

## 21. Специальные типы контекстно-свободных языков и грамматик

**Определение 38.** Говорят, что КС-грамматика  $G = (V_N, V_T, P, S)$  *линейна*, если каждое её правило имеет вид  $A \rightarrow uBv$  или  $A \rightarrow u$ , где  $A, B \in V_N$ ,  $u, v \in V_T^*$ .

Если  $v = \varepsilon$ , то грамматика называется *праволинейной*, если  $u = \varepsilon$ , то она *леволинейна*.

**Определение 39.** Говорят, что грамматика  $G = (V_N, V_T, P, S)$  *последовательна*, если нетерминалы  $A_1, \dots, A_k \in V_N$  можно упорядочить так, что если  $A_i \rightarrow \alpha \in P$ , то  $\alpha$  не содержит ни одного нетерминала  $A_j$  с индексом  $j < i$ .

**Определение 40.** Если КС-язык  $L$  над алфавитом  $V_T$  есть подмножество языка  $w_1^* w_2^* \dots w_k^*$  для некоторого  $k$ , где  $w_i \in V_T^+$ ,  $i = 1, \dots, k$ , то говорят, что  $L$  — *ограниченный язык*.

**Определение 41.** Говорят, что контекстно-свободная грамматика  $G$  *неоднозначна*, если в языке  $L(G)$  существует цепочка, выводимая двумя или более различными левосторонними выводами, то есть существуют различные деревья вывода с одинаковыми результатами.

**Определение 42.** Если все грамматики, порождающие некоторый контекстно-свободный язык, неоднозначны, то говорят, что этот язык *существенно неоднозначен*.

## V. Магазинные автоматы

### 22. МП-автомат. Неформальное описание. Формальное определение. Понятие конфигурации

**Неформальное описание.** Магазинный автомат подобен конечному автомату, но, в отличие от последнего, имеет рабочую память — *магазин*, в который записываются символы из ещё одного алфавита — *алфавита магазинных символов*. Каждое движение магазинного автомата определяется в зависимости от текущего состояния управления, входного символа, или независимо от него, и от верхнего символа магазина.

**Определение 43.** Недетерминированный магазинный автомат есть формальная система  $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ , где

- $Q$  — конечное множество состояний;
- $\Sigma$  — конечный входной алфавит;
- $\Gamma$  — конечный магазинный алфавит;
- $\delta$  — отображение  $Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$ , представляющее конечное управление автомата;
- $q_0 \in Q$  — конечное состояние;
- $Z_0 \in \Gamma$  — начальный символ магазина;
- $F \subset Q$  — множество конечных состояний.

**Определение 44.** Под конфигурацией будем подразумевать тройку  $(q, x, \alpha)$ , где

- $q \in Q$  — текущее состояние управления;
- $x \in \Sigma^*$  — непросмотренная часть входной цепочки;
- $\alpha \in \Gamma^*$  — магазинная цепочка, причём крайний левый её символ считается находящимся на вершине магазина.

### 23. Теорема об эквивалентности языков, принимаемых недетерминированными магазинными автоматами при конечном состоянии и при пустом магазине

**Определение 45.** Язык, распознаваемый магазинным автоматом  $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  при конечном состоянии, определим как множество

$$T(M) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash_M^* (q, \varepsilon, \alpha), \quad q \in F \}$$

**Определение 46.** Язык, распознаваемый магазинным автоматом  $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  при пустом магазине, определим как множество

$$N(M) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash_M^* (q, \varepsilon, \varepsilon), \quad q \in Q \}$$

**Теорема 26.** Язык  $L = N(M_1)$  для некоторого недетерминированного магазинного автомата  $M_1$  тогда и только тогда, когда  $L = T(M_2)$  для некоторого недетерминированного магазинного автомата  $M_2$ .

## 24. Эквивалентность недетерминированных магазинных автоматов и контекстно-свободных грамматик

**Определение 47.** Магазинный автомат  $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  является *детерминированным*, если

1. для любых  $q \in Q$ ,  $Z \in \Gamma$  и  $\alpha \in (\Sigma \cup \{\varepsilon\})$  значение  $\#\delta(q, \alpha, Z) \leq 1$ ;
2. для любых  $q \in Q$  и  $Z \in \Gamma$  всякий раз, как множество  $\delta(q, \varepsilon, Z) \neq \emptyset$ , множество  $\delta(q, a, Z) = \emptyset$  для всех  $a \in \Sigma$ .

**Замечание.** Условие 1 означает, что если движение определено, то оно единственно. Условие 2 предотвращает выбор между  $\varepsilon$ -движением и движением, использующим входной символ.

**Теорема 27.** Если  $L$  — КС-язык, то существует недетерминированный магазинный автомат  $M$  такой, что  $L = N(M)$ .

**Теорема 28.** Если  $M$  — недетерминированный магазинный автомат, и  $L = N(M)$ , то  $L$  — контекстно-свободный язык.

## VI. КСР-грамматики

В этой главе  $V = \{\xi_1, \dots, \xi_n\}$  — непустой алфавит.

### 25. Обобщённые регулярные выражения. КС-грамматика в регулярной форме (КСР-грамматика). Примеры

Этого нигде нет...

### 26. Определение граф-схемы. Лемма о множестве слов $L(\alpha, \beta)$ . Следствия о множестве слов, порождаемых маршрутами в граф-схеме

**Определение 48.** *Граф-схемой* над алфавитом  $V$  называется конечная совокупность ориентированных графов  $\Gamma = \{\Gamma_1, \dots, \Gamma_k\}$ ,  $k \leq n$ , удовлетворяющая следующим условиям:

1. в каждом графе  $\Gamma_i$  выделены две вершины — входная и выходная, причём во входную вершину не входит, а из выходной вершины не выходит ни одна из дуг, а каждая из вершин, отличных от этих двух, помечена буквой из алфавита  $V$ ;
2. графы  $\Gamma_i$  для  $i = 1, \dots, k$  не имеют между собой общих вершин и называются *компонентами граф-схемы*.

**Обозначение.** Входную и выходную вершины в каждом графе  $\Gamma_i$  обозначим  $E_i$  и  $F_i$  соответственно.

**Определение 49.** Буква  $\xi \in V$ , которой помечена вершина  $\alpha \in \Gamma_i$ , называется *меткой вершины*  $\alpha$  и обозначается  $m(\alpha)$ .

**Определение 50.** Каждой неконечной вершине  $\alpha \neq F_i$  поставим в соответствие множество вершин, *смежных* с данной вершиной:

$$\text{succ}(\alpha) = \{\beta \in \Gamma \mid \text{существует дуга из } \alpha \text{ в } \beta\}$$

**Определение 51.** Множество слов  $L_i(\alpha, \beta)$ , порождаемых маршрутами от вершины  $\alpha$  к вершине  $\beta$ , — это

1. пустое слово, если  $\alpha = \beta$ ;
2. слово  $x\xi \in L_i(\alpha, \beta)$ , если  $x \in L_i(\alpha, \gamma)$ , и существует вершина  $\beta$  графа  $\Gamma_i$  такая, что  $\beta \in \text{succ}(\gamma)$  и  $m(\beta) = \xi$ .

**Лемма 7.** Слово  $x \in V^*$  принадлежит множеству слов  $L_i(\alpha, \beta)$  тогда и только тогда, когда существует последовательность вершин  $\alpha_0, \dots, \alpha_k$  графа  $\Gamma_i$  такая, что  $\alpha = \alpha_0$ ,  $\beta = \alpha_k$ ,  $\alpha_i \in \text{succ}(\alpha_{i-1})$  и  $x = \xi_1 \dots \xi_k$ .

**Следствие.** Если  $x \in L_i(\alpha, \beta)$  и  $y \in L_i(\beta, \gamma)$ , то  $xy \in L_i(\alpha, \gamma)$ .

**Следствие.** Если для двух неконечных вершин  $\alpha$  и  $\beta$  графа  $\Gamma_i$  любой маршрут, которым вершина  $\alpha$  соединяется с вершиной  $\beta$ , проходит через вершину  $\gamma$ , то  $L_i(\alpha, \beta) = L_i(\alpha, \gamma) \cdot L_i(\gamma, \beta)$ .

**Обозначение.**  $L_i$  — множество слов, порождаемых всеми маршрутами в графе  $\Gamma_i \in \Gamma$ , начинающихся из входной вершины  $E_i$ :

$$L = \bigcup_{F_i \in \text{succ}(\beta)} (E_i, \beta)$$

**Следствие.** Слово  $x$  в алфавите  $V$  принадлежит множеству  $L_i$  тогда и только тогда, когда существует последовательность вершин графа  $\Gamma_i$   $\alpha_0, \dots, \alpha_k$  такая, что  $\alpha_0 = E_i$ ,  $F_i \in \text{succ}(\alpha_k)$ ,  $\alpha_i \in \text{succ}(\alpha_{i-1})$ ,  $m(\alpha_i) = \xi_i$  и  $x = \xi_1 \dots \xi_k$ .

## 27. Однозначные регулярные выражения. Утверждения об однозначности регулярных выражений

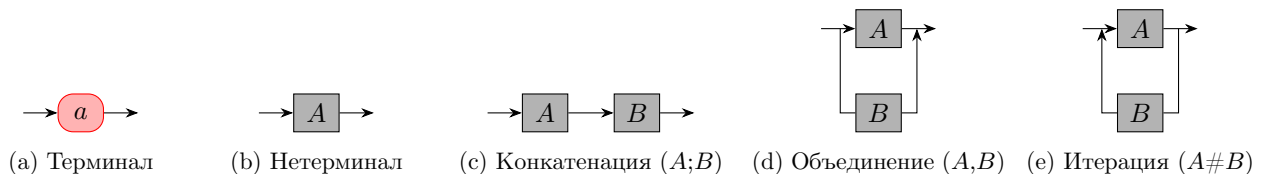
**Определение 52.** Регулярное выражение  $A$  называется *однозначным*, если для любого слова  $x \in L(A)$  в графе  $\Gamma_A$  существует только одна цепочка вершин.

**Утверждение 2.**

1. Все регулярные выражения, представляющие элементарные множества слов, однозначны.
2. Если регулярные выражения  $A$  и  $B$  однозначны, не содержат операции итерации, и оба регулярных множества, которые они представляют, не содержат пустое слово, то регулярное выражение  $C = A;B$  однозначно.
3. Если регулярные выражения  $A$  и  $B$  однозначны, и  $L(A) \cap L(B) = \emptyset$ , то регулярное выражение  $C = A;B$  однозначно.

## 28. Построение графа для нетерминала в КСР-грамматике. Рекуррентный алгоритм построения синтаксической граф-схемы. Лемма 8.2

Будем в дальнейшем алфавитом  $V$  считать объединение алфавитов нетерминалов, терминалов и пустой буквы:  $V = N \cup T \cup \{\lambda\}$ .



Тут есть замудрённые словесные описания, но в целом всё понятно по картиночкам.

**Лемма 8.**  $A$  — регулярное выражение над алфавитом  $V$ ,  $\Gamma_A$  — граф для регулярного выражения  $A$ , построенный по выше описанным требованиям.

Тогда

1.  $L_A = A$ ;
2. во входную вершину  $\Gamma_A$  не входит ни одна дуга, а из выходной вершины  $\Gamma_A$  не выходит ни одна дуга.

**Определение 53.**  $R_A$  — КСР-правило в  $G(N, T, P, S)$ , где

- $A$  — нетерминал из  $N$ ;
- $R_A$  — регулярное выражение над  $V$ ;
- $\Gamma_A$  — граф для регулярного выражения  $R_A$ ;
- $E_A, F_A$  — входная и выходная вершины.

Графом  $\Gamma_A$  для нетерминала  $A \in N$  будем называть граф для регулярного выражения  $R_A$ , в котором входной и выходной вершинам  $E_A$  и  $F_A$  приписаны метки “начало- $A$ ” и “конец- $A$ ”.

**Определение 54.** Синтаксической граф-схемой для КСР-грамматики  $G$  называется совокупность всех графов для нетерминалов из  $N$ , то есть  $\Gamma_G = \{ \Gamma_S, \Gamma_{A_1}, \dots, \Gamma_{A_k} \}$ , где  $S, A_i \in N$ .

## 29. Достижимые вершины. Определение множества начальных вершин $H(A)$ и множества конечных вершин $K(A)$

**Определение 55** (множество начальных вершин).

1. Если  $\alpha$  — вершина графа  $\Gamma_A$  и  $\alpha \in \text{succ}(E_A)$ , то  $\alpha$  — начальная вершина графа  $\Gamma_A$  для нетерминала  $A$ ,  $\alpha \in H(A)$ .
2. Если  $\alpha \in H(A)$  — нетерминальная вершина в  $\Gamma_G$ ,  $m(\alpha) = B$ , и  $\beta \in H(B)$ , то  $\beta$  — начальная вершина графа  $\Gamma_A$ ,  $\beta \in H(A)$ .

**Определение 56** (множество конечных вершин).

1. Если  $\alpha$  — вершина графа  $\Gamma_A$  и выходная вершина  $F_A \in \text{succ}(\alpha)$ , то  $\alpha$  — конечная вершина графа  $\Gamma_A$  для нетерминала  $A$ ,  $\alpha \in K(A)$ .
2. Если  $\alpha \in K(A)$  — нетерминальная вершина в  $\Gamma_G$ ,  $m(\alpha) = B$  и  $\beta \in K(B)$ , то  $\beta$  — конечная вершина графа  $\Gamma_A$ ,  $\beta \in K(A)$ .

## 30. Понятие достижимости на множестве терминальных вершин. Отношение эквивалентности. Лемма и следствие о языке, порождаемом СГС

### Понятие достижимости

**Определение 57.** Между терминальными вершинами  $\alpha_1$  и  $\alpha_2$  в синтаксической граф-схеме  $\Gamma_G$  существует *путь* (*маршрут*), если выполняется одно из условий:

1. Вершины  $\alpha_1$  и  $\alpha_2$  — смежные, принадлежат одной компоненте, и  $\alpha_2 \in \text{succ}(\alpha_1)$  (рис. 2а).

Тогда *путь* от вершины  $\alpha_1$  к вершине  $\alpha_2$  имеет вид  $P_{\alpha_1 \alpha_2} = \{ \alpha_2 \}$ .

2. Существует нетерминальная вершина  $\beta$ , смежная с вершиной  $\alpha_1$ ,  $\beta \in \text{succ}(\alpha_1)$  такая, что  $m(\beta) = B$ , а терминальная вершина  $\alpha_2$  — начальная вершина в графе для нетерминала  $B$ ,  $\alpha_2 \in H(B)$ .

В этом случае *путь* от вершины  $\alpha_1$  к вершине  $\alpha_2$  имеет вид  $\{\beta\} \cdot P_\beta$ , где

- (а) если  $\alpha_2$  принадлежит графу  $\Gamma_B$ , то есть,  $\alpha_2 \in \text{succ}(E_B)$ , то  $P_\beta = P_{E_B \alpha_2} = \{\omega_H \alpha_2\}$ , где  $\omega_H$  — специальный символ, обозначающий *след* прохождения через входную вершину графа для нетерминала (рис. 2b);
- (б) если в графе  $\Gamma_B$  существует нетерминальная вершина  $\gamma \in \Gamma_B$  такая, что  $m(\gamma) = C$ ,  $\gamma \in \text{succ}(E_B)$  и  $\alpha_2 \in H(C)$ , то  $P_\beta = P_{E_B \gamma} \cdot P_\gamma = \{\omega_H \gamma\} \cdot P_\gamma$  (рис. 2c).

3.  $\alpha_1$  — конечная вершина графа  $\Gamma_B$  и  $\alpha_2 \in \text{succ}(\beta)$ , где  $\beta$  — нетерминальная вершина, и  $m(\beta) = B$ .

Тогда *путь* от вершины  $\alpha_1$  к вершине  $\alpha_2$  имеет вид  $P_{\alpha_1 \alpha_2} = P_{\alpha_1}^\beta \cdot \{\alpha_2\}$ , где

- (а) если  $\alpha_1$  — вершина графа  $\Gamma_B$ , т. е.  $F_B \in \text{succ}(\alpha_1)$ , то  $P_{\alpha_1}^\beta = \{\omega_K \beta\}$ , где  $\omega_K$  обозначает *след* прохождения через выходную вершину графа для соответствующего нетерминала (рис. 2d);
- (б) если  $\alpha_2$  — конечная вершина графа  $\Gamma_C$ , т. е.  $F_C \in \text{succ}(\alpha_1)$  и существует нетерминальная вершина  $\gamma$ ,  $m(\gamma) = C$ , и  $\gamma \in K(B)$ , то  $P_{\alpha_1}^\beta = \{\omega_K \gamma\} \cdot P_\gamma^\beta$  (рис. 2e).

4. Вершина  $\alpha_2$  — конечная в графе  $\Gamma_B$ , то есть  $\alpha_1 \in K(B)$ ,  $\alpha_2$  — начальная вершина графа  $\Gamma_C$ , т. е.  $\alpha_2 \in H(C)$ , и существуют нетерминальные вершины  $\beta$  и  $\gamma$  в графе  $\Gamma_A$  такие, что  $m(\beta) = B$  и  $m(\gamma) = C$ .

Тогда *путь* между вершинами  $\alpha_1$  и  $\alpha_2$  имеет вид  $P_{\alpha_1 \alpha_2} = P_{\alpha_1}^\beta \cdot \{\gamma\} \cdot P_\gamma$  (рис. 2f).

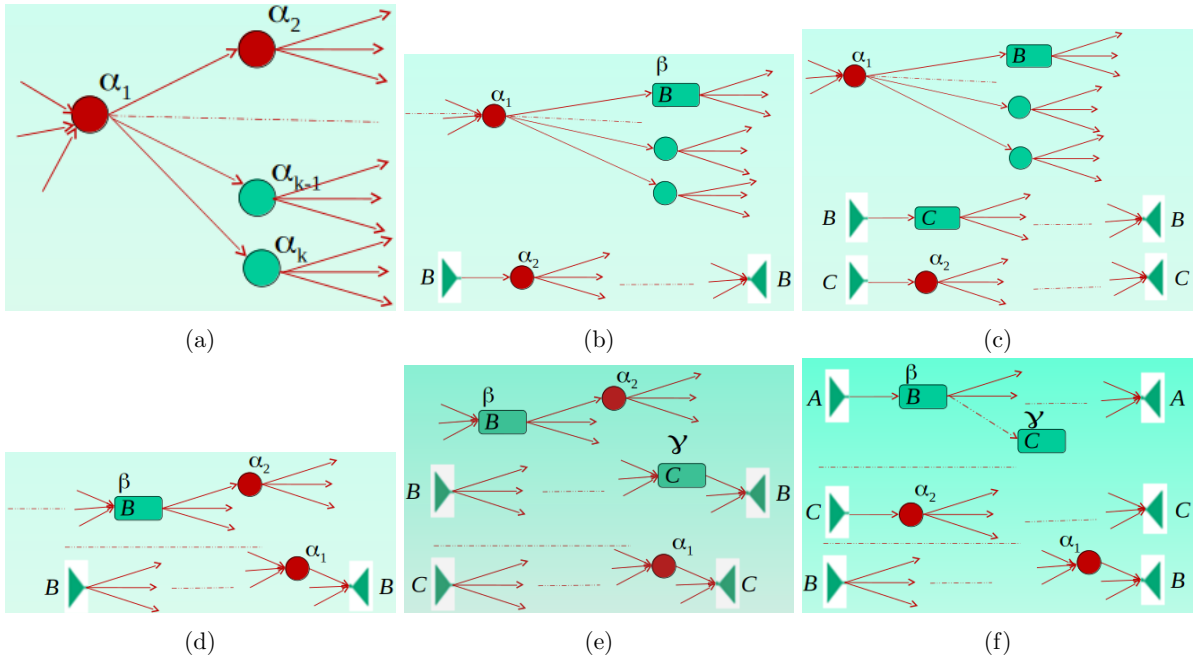


Рис. 2: К определению пути

**Определение 58.** Терминальная вершина  $\alpha_2$  *достижима* из терминальной вершины  $\alpha_1$  в синтаксической граф-схеме  $\Gamma_G$ , если существует путь  $P_{\alpha_1 \alpha_2}$ .

**Обозначение.**  $\alpha_1 \xrightarrow{G} \alpha_2$

### Отношение эквивалентности

Рассмотрим множество всех вершин в синтаксической граф-схеме  $\Gamma_G$ . Исключим из него все входные и выходные вершины компонент графов и добавим два новых элемента  $\omega_H$  и  $\omega_K$ , где  $\omega_H$  заменяет все входные, а  $\omega_K$  — выходные вершины. Полученное множество обозначим  $\mathfrak{Z}$ .



**Определение 59.**  $\alpha, \beta$  — нетерминальные вершины в множестве  $\mathfrak{V}$ ,  $x$  — цепочка терминальных вершин,  $X, Y$  — произвольные цепочки в алфавите  $\mathfrak{V} \cup \{\Omega\}$ .

Введём на множестве всех маршрутов синтаксической граф-схемы *отношение эквивалентности*:

$$X\omega_H x\omega_K Y \sim XxY, \quad X\alpha\omega_H x\omega_K \beta Y \sim \begin{cases} XxY, & \alpha = \beta, \\ \Omega, & \end{cases} \quad X\Omega Y \sim \Omega, \quad X \sim X$$

Двум терминальным вершинам  $\alpha$  и  $\beta$ , принадлежащим синтаксической-граф схеме  $\Gamma_G$ , поставим в соответствие множество слов  $L_{\Gamma_G}(\alpha, \beta)$ , определяемое следующим образом:

1.  $\varepsilon \in L_{\Gamma_G}(\alpha, \beta)$ , если  $\alpha = \beta$ ;
2. если  $x \in L_{\Gamma_G}(\alpha, \gamma)$ , существует вершина  $\beta$  такая, что  $\gamma \xrightarrow{G} \beta$ ,  $m(\beta) = \xi$ , и существует путь  $P_{\gamma\beta}$  такой, что  $P_x \cdot P_{\gamma\beta} \neq \Omega$ , то  $x\xi \in L_{\Gamma_G}(\alpha, \beta)$ .

$$L_{\Gamma_G}(E_A) := \bigcup_{\beta \in K(A)} L_{\Gamma_G}(E_A, \beta)$$

Если  $A = S$ , то  $L_{\Gamma_G}(E_S)$  будем обозначать  $L_{\Gamma_G}$ .

**Лемма 9.** Слово  $x$  в алфавите терминалов  $T$  принадлежит множеству  $L_{\Gamma_G}(\alpha, \beta)$  **тогда и только тогда**, когда существует последовательность терминальных вершин  $\alpha_0, \dots, \alpha_k$ , и последовательность путей  $P_1, \dots, P_k$  в СГС  $\Gamma_G$  такие, что

$$\alpha_0 = \alpha, \quad \alpha_k = \beta, \quad m(\alpha_i) = \xi_i, \quad \alpha_{i-1} \xrightarrow{G} \alpha_i, \quad P_i \sim P_{\alpha_{i-1}\alpha_i}, \quad x = \xi_1, \dots, \xi_k, \quad P_x \sim P_1 \cdots P_k, \quad P_x \neq \Omega$$

**Следствие.** Слово  $x$  в алфавите  $T$  принадлежит множеству  $L_G(E_A)$  **тогда и только тогда**, когда существует последовательность терминальных вершин  $\alpha_0, \dots, \alpha_k$  в  $\Gamma_G$  и последовательность путей  $P_1, \dots, P_k$  в  $\Gamma_G$  такие, что

$$\alpha_0 = E_A, \quad \alpha_k \in K(A), \quad \alpha_{i-1} \xrightarrow{G} \alpha_i, \quad m(\alpha_i) = \xi_i, \quad P_i \sim P_{\alpha_{i-1}\alpha_i}, \quad x = \xi_1, \dots, \xi_k, \quad P_x \sim P_1 \cdots P_k,$$

причём существуют такие вершины  $\beta_1, \dots, \beta_l$  в  $\Gamma_G$ , что  $P_x \cdot \omega_k \beta_1 \cdot \omega_k \beta_2 \cdots \omega_k \beta_l = \alpha_1 \dots \alpha_k$ .

## 31. Синтез распознающего автомата для КСР-грамматики. Состояние для регулярного выражения в графе $\Gamma_A$ . Состояние вершины в графе $\Gamma_A$ . Переходное состояние. Регулярный случай

Рассмотрим случай, когда КСР-грамматика  $G$  порождает регулярный язык, а множество  $P$  правил грамматики содержит только одно  $S$ -правило для начального нетерминала  $S$ ,  $P = \{S : A\}$ . Синтаксическая граф-схема  $\Gamma_G$  состоит из одного графа  $\Gamma_G = \{\Gamma_S\}$  и не содержит нетерминальных вершин.

**Определение 60.** *Состоянием (для регулярного выражения) в графе  $\Gamma_A$  называется:*

- выходная вершина  $F_A$ ;
- терминальная вершина в  $\Gamma_A$ ;
- объединение состояний в  $\Gamma_A$ .

**Определение 61.** *Состоянием вершины  $\beta$  в графе  $\Gamma_A$  называется множество вершин*

$$S_\beta = \{ \alpha \mid \alpha \text{ — терм. или вых. вершина в } \Gamma_A, \quad \alpha \in \text{succ}(\beta) \}$$

Если  $\beta$  — начальная вершина графа  $\Gamma_A$ , то  $S_\beta$  называется *начальным состоянием* графа  $\Gamma_A$ .

**Следствие.** Для любого слова  $x \in L_A$ ,  $x = \xi_1 \dots \xi_k$  существует путь порождения этого слова в графе  $\Gamma_A$  такой, что

$$\alpha_0, \dots, \alpha_k \in P_x, \quad \alpha_{i-1} \rightarrow \alpha_i, \quad \alpha_0 = E_A, \quad \alpha_1 \in S_{E_A}, \quad \alpha_i \in S_{\alpha_{i-1}}, \quad F_A \in S_{\alpha_k}, \quad m(\alpha_i) = \xi_i$$

**Определение 62.**  $S$  — состояние в графе  $\Gamma_A$ ,  $\xi$  — буква из алфавита  $T$ .

Состояние  $S/\xi$  называется *переходом по символу* (*переходным состоянием* для  $S$ ) в граф-схеме  $\Gamma_A$ , причём

- если  $S = \emptyset$  или  $S = \{F\}$ , то  $S/\xi = \emptyset$ ;
- если  $S = \{\alpha\}$ ,  $\alpha$  — терминальная вершина, то

$$S/\xi = \begin{cases} \emptyset, & m(\alpha) \neq \xi, \\ S_\alpha, & m(\alpha) = \xi; \end{cases}$$

- если  $S = \bigcup_{i=1}^k S_i$ , то  $S/\xi = \bigcup_{i=1}^k S_i/\xi$ .

**Определение 63.**  $S$  — состояние в графе  $\Gamma_A$ ,  $x = x'\xi$  — слово в алфавите  $T$ ,  $x' \in T^*$ .

*Переходом по слову  $x$*  называется состояние  $S/x = (S/x')/\xi$ .

*Переходом по пустому слову  $\varepsilon$*  называется состояние  $S/\varepsilon = S$ .

**Утверждение 3.** Язык, порождаемый графом для регулярного выражения, состоит из тех слов, переход по которым для начального состояния  $S_{E_A}$  данного графа содержит выходную вершину:

$$L_A = \left\{ x \in T^* \mid F_A \in S_{E_A/x} \right\}$$

## 32. Синтез распознающего автомата для КСР-грамматики. Состояния в синтаксической граф-схеме. Состояние вершины в СГС. Переходное состояние. Общий случай

КСР-грамматика порождает КС-язык, а регулярные выражения для КСР-правил содержат нетерминалы. Синтаксическая граф-схема состоит из совокупности графов  $\Gamma_G = (\Gamma_S, \Gamma_{A_1}, \dots, \Gamma_{A_k})$  и содержит нетерминальные вершины.

**Определение 64.** *Состоянием* в граф-схеме  $\Gamma_G$  называется

- выходная вершина  $F_{A_i}$  графа  $\Gamma_{A_i}$  для некоторого нетерминала  $A_i \in N$ ;
- терминальная вершина в  $\Gamma_G$ ;
- пара множеств  $(S_1, S_2)$ , где  $S_1$  и  $S_2$  — состояния в  $\Gamma_G$ ;
- объединение состояний в  $\Gamma_G$ .

**Определение 65.** Для вершины  $\beta$  в СГС  $\Gamma_G$  состояние вершины  $\beta$  имеет вид:

$$S_\beta = \left\{ \begin{array}{l} \alpha \mid \alpha \in \text{succ}(\beta), \quad \alpha \text{ — терм. или вых. вершина в } \Gamma_G \\ (S_{E_{m(\alpha)}}, S_\alpha) \mid \alpha \in \text{succ}(\beta), \quad \alpha \text{ — нетерм. вершина в } \Gamma_G \end{array} \right\}$$

**Определение 66.** Если  $\beta$  — входная вершина графа  $\Gamma_{A_i}$  для некоторого нетерминала  $A_i \in N$ , то состояние  $S_\beta$  в  $\Gamma_G$  будем называть *начальным состоянием* графа  $G_{A_i}$ .

**Обозначение.**  $S_{E_{A_i}}$

**Определение 67.** Начальное состояние для стартового символа КСР-грамматики называется *начальным состоянием* СГС  $\Gamma_G$ .

**Обозначение.**  $S_0$

Состояние в  $\Gamma_G$  — это, в общем случае, объединение цепочек вложенных пар множеств терминальных или выходных вершин. Всё множество состояний в  $\Gamma_G$  можно разбить на классы.

Обозначим через  $\mathfrak{Z}_k$  множество состояний  $k$ -го уровня.

**Определение 68.** Состояние  $S$  принадлежит множеству  $\mathfrak{Z}_0$ , если

- $S = \{ \alpha \}$ , где  $\alpha$  — терминальная или выходная вершина;
- $S = \bigcup S_i$ , где  $S_i \in \mathfrak{Z}_0$ .

**Определение 69.** Состояние  $S$  принадлежит множеству  $\mathfrak{Z}_k$ , если

- $S = (S_1, S_2)$ , где  $S_1, S_2 \in \bigcup_{i=1}^k S_i$ ;
- $S = \bigcup_{i=1}^k S_i$ , где  $S_i \in \mathfrak{Z}_k$ .

Процесс построения состояния для некоторой вершины может не завершиться. В таком случае будем считать, что состояние этой вершины не существует. Множество всех состояний, которые существуют в  $\Gamma_G$ , обозначим  $\mathfrak{Z}_G$ .

**Определение 70.** Состояние  $S/\xi$  называется *переходным состоянием* по символу  $\xi$  для состояния  $S$  в  $\Gamma_G$ , причём

- если  $S = \emptyset$  или  $S = \{ F_{A_i} \}$ , то  $S/\xi = \emptyset$ ;
- если  $S = \{ \alpha \}$ , где  $\alpha$  — терминальная вершина, то

$$S/\xi = \begin{cases} \emptyset, & m(\alpha) \neq \xi, \\ S_\alpha, & m(\alpha) = \xi; \end{cases}$$

- если  $S = \bigcup S_i$ , то  $S/\xi = \bigcup (S_i/\xi)$ ;
- если  $S = (S_1, S_2)$ , то

$$X/\xi = \begin{cases} (S_1/\xi, S_2), & S_1/\xi \neq \emptyset, \\ S_2/\xi, & S_1/\xi = \emptyset, \quad F_a \in S_1, \\ \emptyset. & \end{cases}$$

### 33. Свойства синтаксической граф-схемы. Леммы о существовании состояний распознавателя в синтаксической граф-схеме

**Определение 71.** Нетерминальная вершина  $\beta$  в  $\Gamma_G$  обладает свойством **D**, если множество состояний  $\mathfrak{Z}_G$  не содержит состояния вида  $(\dots (\{ F_{m(\beta)}, S \}, S_\beta) \dots)$ , в котором “фактическое” состояние в  $S$  содержит терминальную вершину  $\alpha$ , а состояние  $S_\beta$  содержит вершину  $\gamma$  и  $m(\alpha) = m(\gamma)$ .

Свойство **D** гарантирует, что состояние вершины  $\beta$  не создаёт неопределённости при распознавании языка (*конфликт типа Shift/Reduce*).

**Определение 72.** Нетерминальная вершина  $\beta$  в  $\Gamma_G$  обладает свойством **D<sub>1</sub>**, если

1. вершина  $\beta$  обладает свойством **D**;
2. множество начальных вершин  $H(m(\beta))$  не содержит нетерминальной вершины  $\gamma \in \text{succ}(\beta)$ .

Свойство **D<sub>1</sub>** показывает, что состояние терминальной вершины  $\alpha$  такой, что  $\beta \in \text{succ}(\alpha)$ , принадлежит классу  $\mathfrak{Z}_1$ .

**Определение 73.** Для  $k > 1$  считаем, что нетерминальная вершина  $\beta$  в синтаксической граф-схеме  $\Gamma_G$  обладает свойством  $\mathbf{D}_k$ , если

1. вершина  $\beta$  обладает свойством  $\mathbf{D}$ ;
2.
  - либо существует нетерминальная вершина  $\gamma \in \text{succ}(\beta)$ , вершина  $\gamma$  обладает свойством  $\mathbf{D}_t$  ( $1 \leq t < k$ ), и все начальные нетерминальные вершины из множества  $H(m(\beta))$  обладают свойством  $\mathbf{D}_t$  ( $1 \leq t < k$ );
  - либо не существует такой вершины  $\gamma \in \Gamma_G$ , и среди начальных вершин из множества  $H(m(\beta))$  есть хотя бы одна нетерминальная вершина, обладающая свойством  $\mathbf{D}_{k-1}$ , а все другие нетерминальные вершины обладают свойством  $\mathbf{D}_j$  для  $j < k-1$ .

Будем считать, что СГС обладает свойством  $\mathbf{D}_k$ , если существует хотя бы одна вершина, обладающая свойством  $\mathbf{D}_k$ .

**Лемма 10.** Для любой нетерминальной вершины  $\beta$  в  $\Gamma_G$ , обладающей свойством  $\mathbf{D}_k$ ,  $m(\beta) = B$ , состояние  $(S_{E_B}, S_\beta) \in \mathfrak{S}_k$ .

**Лемма 11.** Для любой нетерминальной вершины  $\beta$  в  $\Gamma_G$ , обладающей свойством  $\mathbf{D}_k$ ,  $m(\beta) = B$  начальные состояние  $S_{E_B}$  существует,  $S_{E_B} \in \mathfrak{S}_t$  ( $0 \leq t < k$ ), и для всякой терминальной начальной вершины  $\alpha \in H(B)$  существует последовательность нетерминальных вершин  $\beta_1, \dots, \beta_l$  ( $0 \leq l < k$ ) в  $\Gamma_G$  таких, что

$$\begin{cases} \forall 1 \leq i \leq l & m(\beta_i) = B_i, \\ \forall 2 \leq i \leq l & (S_{E_{B_i}}, S_{\beta_i}) \in S_{E_{B_{i-1}}}, \\ \left( \dots ((S_{E_{B_l}}, S_{\beta_l}), S_{\beta_{l-1}}), \dots, S_{\beta_1} \right) \in S_{E_B}, \\ \alpha \in S_{E_{B_l}}, \\ P_\beta = \omega_H \beta_1 \omega_H \beta_2 \dots \omega_H \beta_l \omega_H \alpha. \end{cases}$$

## 34. Регуляризация КС-грамматики. Эквивалентные преобразования. Базисные преобразования. Синтаксическая модель языка

### Синтаксическая модель языка

*Модель языка* — это способ его описания. *Синтаксическая модель* предполагает четыре аспекта:

1. *Лексика* определяет представление основных символов языка.
2. *Синтаксис* определяет представление основных конструкций языка посредством терминальных символов.
3. Не все нетерминалы порождают конструкции, а только те, для которых определена *семантика*.
4. *Прагматика*.

### Базисные преобразования

- Подстановка вместо нетерминала его порождения;
- удаление вхождения лево-(право-)рекурсивного нетерминала в правой части правила;
- объединение общих префиксов в графе для нетерминала;
- удаление повторяющихся альтернатив для нетерминала;
- удаление крайних рекурсий для самовложенных нетерминалов;
- свёртка регулярного подвыражения в новый нетерминал;
- удаление лишних правил.

### 35. Алгоритм исключения лево-(право-)рекурсивных нетерминалов в КСР-правиле. Общий случай для всех правил КСР-грамматики

Рассмотрим  $A$  — правило в КСР-грамматике, когда нетерминал является одновременно и лево-, и праворекурсивным, и рекурсия прямая:

$$A : A, r_{11}, A; A, r_{12}; r_{21}, A; r_{22},$$

где  $r_{11}, r_{12}, r_{21}, r_{22}$  — регулярные выражения.

#### Алгоритм.

1. Рассмотрим  $A_1$ -фрагмент  $A, r_{11}, A$ . Используя данный фрагмент, мы можем вывести строки

$$Ar_{11}A, \quad Ar_{11}Ar_{11}A, \quad \dots$$

Из определения операции  $\#$ , это множество строк совпадает с множеством строк, порождаемым регулярным выражением  $A\#r_{11}$ .

2. Рассмотрим  $A_2$ -фрагмент:  $A, r_{12}$ . Можем вывести строки:  $Ar_{12}, Ar_{12}r_{12}, \dots$  Из определения операции  $*$  следует, что это множество порождается регулярным выражением  $A(r_{12})^*$ .
3. Аналогично для праворекурсивного вхождения нетерминала  $A$  в  $A_3$ -фрагменте. Здесь мы выводим множество строк  $r_{21}A, r_{21}r_{21}A, \dots$ , которые порождаются регулярным выражением  $r_{21}^*A$ .
4. Окончательно, подставляя все регулярные выражения, получаем

$$(r_{21}^*, r_{22}, r_{12}^*)\#r_{11}$$

### 36. Схема получения регулярного выражения, эквивалентного приведённой КС-грамматике без самовставлений. Пример

Этого нет...

## VII. Трансляции, их представление и реализация

### 1. Некоторые способы задания трансляций: перечисление, гомоморфизм, схемы синтаксически-управляемых трансляций, конечные и магазинные преобразователи

**Определение 74.** Трансляцией из языка  $L_1 \subset \Sigma^*$  называется отношение  $\tau \subset L_1 \times L_2$ .

Здесь  $\Sigma$  — входной алфавит,  $L_1$  — входной язык,  $\Delta$  — выходной алфавит,  $L_2$  — выходной язык.

По отношению к языкам программирования, трансляция всегда является функцией.

#### Гомоморфизм

Гомоморфизм  $h$  определяет трансляцию

$$\tau(h) = \{ (x, h(x)) \mid x \in \Sigma^* \}$$

Устройство, которое по заданной цепочке  $x \in \Sigma^*$  находит соответствующую цепочку  $y = h(x)$ , должно посимвольно просмотреть входную цепочку  $x$  и заменить каждый её символ  $a$  на  $h(a)$ .

## Схемы синтаксически-управляемой трансляции

**Определение 75.** Схемой синтаксически-управляемой трансляции называется формальная система  $T = (N, \Sigma, \Delta, R, S)$ , где

- $N$  — алфавит нетерминалов;
- $\Sigma$  — конечный входной алфавит;
- $\Delta$  — конечный выходной алфавит, причём  $N \cap \Sigma = \emptyset$  и  $N \cap \Delta = \emptyset$ ;
- $R$  — конечное множество правил вида  $A \rightarrow \alpha, \beta$ , где
  - $A \in N$ ;
  - $\alpha \in (N \cup \Sigma)^*$ ;
  - $\beta \in (N \cup \Delta)^*$ ;
  - каждое вхождение нетерминала в цепочку  $\alpha$  связано с некоторым вхождением одноимённого нетерминала в  $\beta$ , и эта связь является неотъемлемой частью правила;
- $S \in N$  — начальный нетерминал.

Цепочка  $\alpha$  называется синтаксической, а  $\beta$  — семантической.

**Определение 76.** Введём понятие трансляционной формы:

- $(S, S)$  — начальная трансляционная форма, причём эти два вхождения начального нетерминала связаны друг с другом по определению.
- Если  $(\alpha A \beta, \alpha' A \beta')$  — трансляционная форма, в которой два явно выделенных вхождения нетерминала  $A$  связаны, и если  $A \rightarrow \gamma, \gamma'$  — правило из  $R$ , то  $(\alpha \gamma \beta, \alpha' \gamma' \beta')$  — трансляционная форма. Связь между нетерминалами в  $\gamma$  и  $\gamma'$  такая же, как в правиле. Нетерминалы в цепочках  $\alpha$  и  $\beta$  связываются с нетерминалами в цепочках  $\alpha'$  и  $\beta'$  в новой трансляционной форме так же, как в предыдущей.
- Никакие другие пары цепочек не являются трансляционными формами.

**Обозначение.** Отношение непосредственной выводимости:  $(\alpha A \beta, \alpha' A \beta') \xRightarrow{T} (\alpha \gamma \beta, \alpha' \gamma' \beta')$

**Определение 77.** Трансляция, заданная при помощи схемы синтаксически-управляемой трансляции  $T$  есть множество

$$\tau(T) = \left\{ (x, y) \mid (S, S) \xRightarrow{T}^* (x, y), \quad x \in \Sigma^*, y \in \Delta^* \right\}$$

и называется синтаксически-управляемой трансляцией.

**Определение 78.** Грамматика  $G_i = (N, \Sigma, P_i, S)$ , где  $P_i = \{ A \rightarrow \alpha \mid \exists A \rightarrow \alpha, \beta \in R \}$ , называется входной грамматикой схемы.

Грамматика  $G_0 = (N, \Delta, P_0, S)$ , где  $P_0 = \{ A \rightarrow \beta \mid \exists A \rightarrow \alpha, \beta \in R \}$  называется выходной грамматикой схемы.

## 2. Простые SDTS. Эквивалентность классов трансляций, задаваемых простыми SDTS и недетерминированными магазинными преобразователями

**Определение 79.** Схема синтаксически-управляемой трансляции называется простой, если в каждом её правиле  $A \rightarrow \alpha, \beta$  связанные нетерминалы в цепочках  $\alpha$  и  $\beta$  встречаются в одинаковом порядке.

**Определение 80.** Недетерминированный магазинный преобразователь — это формальная система  $P = (Q, \Sigma, \Gamma, \Delta, \delta, q_0, Z_0, F)$ , где

- $Q$  — конечное множество состояний;
- $\Sigma$  — конечный входной алфавит;
- $\Gamma$  — конечный алфавит магазинных символов;
- $\Delta$  — конечный выходной алфавит;
- $q_0 \in Q$  — начальное состояние;
- $Z_0 \in \Gamma$  — начальный символ магазина;
- $F \subset Q$  — множество конечных состояний;
- $\delta$  — отображение  $Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^* \times \Delta^*}$ .

**Определение 81.** Конфигурацией магазинного преобразователя  $P$  назовём четвёрку  $(q, x, \alpha, y)$ , где

- $Q$  — текущее состояние;
- $x \in \Sigma^*$  — непросмотренная часть входной цепочки;
- $\alpha \in \Gamma^*$  — содержимое магазина;
- $y \in \Delta^*$  — вся выходная цепочка.

**Определение 82.** Говорят, что  $y \in \Delta^*$  — выход для  $x \in \Sigma^*$  при конечном состоянии, если  $(q_0, x, Z_0, \varepsilon) \vdash^* (q, \varepsilon, \alpha, y)$  для некоторых  $q \in F$  и  $\alpha \in \Gamma^*$ .

Трансляция, определяемая магазинным преобразователем  $P$  при конечном состоянии, есть

$$\tau(P) = \{ (x, y) \mid (q_0, x, Z_0, \varepsilon) \vdash^* (q, \varepsilon, \alpha, y), \quad q \in F, \alpha \in \Gamma^* \}$$

**Определение 83.** Говорят, что  $y \in \Delta^*$  есть выход для  $x \in \Sigma^*$  при пустом магазине, если  $(q_0, x, Z_0, \varepsilon) \vdash^* (q, \varepsilon, \varepsilon, y)$  для некоторого  $q \in Q$ .

Трансляция, определяемая магазинным преобразователем  $P$  при пустом магазине, есть

$$\tau_e(P) = \{ (x, y) \mid (q_0, x, Z_0, \varepsilon) \vdash^* (q, \varepsilon, \varepsilon, y), \quad q \in Q \}$$

**Определение 84.** Магазинный преобразователь  $P = (Q, \Sigma, \Gamma, \Delta, \delta, q_0, Z_0, F)$  называется *детерминированным*, если

1.  $\#\delta(q, a, Z) \leq 1 \quad \forall q \in Q, a \in \Sigma \cup \{\varepsilon\}, Z \in \Gamma$ ;
2. если  $\delta(q, \varepsilon, Z) \neq \emptyset$ , то  $\delta(q, a, Z) = \emptyset \quad \forall a \in \Sigma$ .

**Лемма 12.**  $T$  — простая схема синтаксически-управляемой трансляции.

Существует недетерминированный магазинный преобразователь  $P$  такой, что  $\tau_e(P) = \tau(T)$ .

**Лемма 13.**  $P$  — недетерминированный магазинный преобразователь.

Существует простая схема синтаксически-управляемой трансляции  $T$  такая, что  $\tau(T) = \tau_e(P)$ .

Есть ещё теорема, которая две леммы зачем-то объединяет в “тогда и только тогда”. Я позволю себе здесь её не приводить.

### 3. Эквивалентность классов трансляций, задаваемых магазинными преобразователями при конечном состоянии и при пустом магазине. Теорема 1.2

**Лемма 14.**  $P$  — недетерминированный магазинный преобразователь и  $\tau = \tau(P)$ .

Существует недетерминированный магазинный преобразователь  $P'$  такой, что  $\tau_e(P') = \tau$ .

**Лемма 15.**  $P$  — недетерминированный магазинный преобразователь и  $\tau = \tau_e(P)$ .

Существует недетерминированный магазинный преобразователь  $P'$  такой, что  $\tau(P') = \tau$ .

Теорема 1.2 — это такая же бессмысленная теорема, как и в предыдущем параграфе.

#### 4. Детерминированная генерация выходной цепочки простой SDT по левостороннему анализу входной цепочки. Теорема 1.3

**Определение 85.** Схема синтаксически-управляемой трансляции называется *семантически однозначной*, если в ней не существует двух правил вида  $A \rightarrow \alpha, \beta$  и  $A \rightarrow \alpha, \gamma$ , в которых  $\beta \neq \gamma$ .

**Определение 86.**  $G = (V_N, V_T, P, S)$  — cfg, правила которой пронумерованы от 1 до  $p$ , и  $S \xRightarrow[\text{lm}]{\pi} x$  — левосторонний вывод  $x \in V_T^*$  в грамматике  $G$ .

Последовательность номеров правил  $\pi$ , применённых в этом выводе называется *левосторонним анализом* цепочки  $x$ .

**Теорема 29.**  $T = (N, \Sigma, \Delta, R, S)$  — простая семантически однозначная схема синтаксически-управляемой трансляции, правила которой пронумерованы от 1 до  $p$ .

Существует детерминированный магазинный преобразователь  $P$  такой, что

$$\tau_e(P) = \left\{ (\pi, y) \mid (S, S) \xRightarrow[\text{lm}]{\pi} (x, y) \text{ для некоторой цепочки } x \in \Sigma^* \right\}$$

## VIII. $\text{LL}(k)$ -грамматики и трансляции

#### 5. Определение и свойства $\text{LL}(k)$ -грамматики. Необходимые и достаточные условия принадлежности приведённой КС-грамматики классу $\text{LL}(k)$ . Теорема 2.1

**Определение 87.**  $G = (V_N, V_T, P, S)$  — cfg.

Определим функцию

$$\text{FIRST}_k^G(\alpha) = \left\{ w \in V_T^* \mid \left[ \begin{array}{ll} |w| < k, & \alpha \xRightarrow[G]{*} w, \\ |w| = k, & \alpha \xRightarrow[G]{*} wx \text{ для нек. цепочки } x \in V_T^* \end{array} \right] \right\}$$

Здесь  $k \geq 0$  — целое,  $\alpha \in (V_N \cup V_T)^*$ .

**Определение 88.**  $G = (V_N, V_T, P, S)$  — cfg.

Говорят, что  $G$  есть  $\text{LL}(k)$ -грамматика для некоторого фиксированного  $k$ , если для любых двух левосторонних выводов вида

$$1. S \xRightarrow[\text{lm}]{*} wA\alpha \xRightarrow[\text{lm}]{*} w\beta\alpha \xRightarrow[\text{lm}]{*} wx,$$

$$2. S \xRightarrow[\text{lm}]{*} wA\alpha \xRightarrow[\text{lm}]{*} w\gamma\alpha \xRightarrow[\text{lm}]{*} wy,$$

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

**Определение 89.** Говорят, что cfg есть  $\text{LL}$ -грамматика, если она  $\text{LL}(k)$  для некоторого  $k$ .

**Определение 90.** Говорят, что cfg  $G$  является *простой*  $\text{LL}(1)$ -грамматикой, если в ней нет  $\varepsilon$ -правил, и все альтернативы для каждого нетерминала начинаются с терминалов и притом различных.



**Теорема 30.** Чтобы cfg  $G$  была  $LL(k)$ -грамматикой, **необходимо и достаточно**, чтобы

$$FIRST_k^G(\beta\alpha) \cap FIRST_k^G(\gamma\alpha) = \emptyset$$

для всех  $\alpha, \beta, \gamma$  таких, что существуют правила  $A \rightarrow \beta$ ,  $A \rightarrow \gamma$ ,  $\beta \neq \gamma$  и существует вывод  $S \xRightarrow{*}_{lm} wA\alpha$ .

## 6. Алгоритм вычисления функции $FIRST_k^G(\beta)$ и его обоснование

Этого тоже нет...

## 7. Определение функции $FOLLOW_k^G(\beta)$

**Определение 91.**  $G = (V_N, V_T, P, S) - \text{cfg}$ ,  $\beta \in V^*$

Определим функцию

$$FOLLOW_k^G(\beta) = \left\{ w \in V_T^* \mid S \xRightarrow{*}_G \gamma\beta\alpha, \quad w \in FIRST_k^G(\alpha) \right\}$$

Здесь  $k \geq 0$  — целое.

## 8. Необходимые и достаточные условия принадлежности приведённой КС-грамматики классу $LL(1)$ . Сильные $LL(k)$ -грамматики. Теорема 2.2

**Теорема 31.** Чтобы cfg  $G = (V_N, V_T, P, S)$  была  $LL(1)$ -грамматикой, **необходимо и достаточно**, чтобы

$$FIRST_1^G(\beta FOLLOW_1^G(A)) \cap FIRST_1^G(\gamma FOLLOW_1^G(A)) = \emptyset$$

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

**Следствие.** КС-грамматика  $G$  является  $LL(1)$ -грамматикой **тогда и только тогда**, когда для каждого множества  $A$ -правил:  $A \rightarrow \alpha_1 | \alpha_2 | \dots | \alpha_n$  — выполняются условия:

1.  $FIRST_1^G(\alpha_i) \cap FIRST_1^G(\alpha_j) = \emptyset$ ;
2. если  $\alpha_i \xRightarrow{*}_G \varepsilon$ , то  $FIRST_1^G(\alpha_j) \cap FOLLOW_1^G(A) = \emptyset$ .

**Определение 92.** КС-грамматика  $G = (V_N, V_T, P, S)$  называется *сильной  $LL(k)$ -грамматикой*, если

$$FIRST_k^G(\beta FOLLOW_k^G(A)) \cap FIRST_k^G(\gamma FOLLOW_k^G(A)) = \emptyset$$

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

**Следствие.** Каждая  $LL(1)$ -грамматика является сильной.

## 9. Достаточные признаки непринадлежности КС-грамматики классу $LL$ . Теорема 2.3

**Теорема 32.** Если  $G$  — cfg и  $G$  леворекурсивна, то  $G$  не  $LL(k)$ -грамматика ни при каком  $k$ .

**Следствие.** Имеем два достаточных признака, чтобы считать КС-грамматику не  $LL$ -грамматикой. Это — неоднозначность и леворекурсивность.

## 10. $k$ -предсказывающие алгоритмы анализа. Формальное определение

**Определение 93.**  $k$ -предсказывающим алгоритмом анализа называется формальная система  $\mathcal{A} = (\Sigma, \Gamma \cup \{ \$ \}, \Delta, M, X_0, \$)$ , где

- $\Sigma$  — входной алфавит;
- $\Gamma \cup \{ \$ \}$  — магазинный алфавит;
- $\$ \notin \Gamma$  — маркер дна магазина;
- $\Delta$  — выходной алфавит;
- $X_0 \in \Gamma$  — начальный символ магазина;
- $M : (\Gamma \cup \{ \$ \}) \times \Sigma^{*k} \rightarrow \{ (\beta, i), \text{pop}, \text{accept}, \text{error} \}$  — управляющая таблица, причём  $\beta \in \Gamma^*$ ,  $i \in \Delta$  — номер правила грамматики.

**Определение 94.** Под конфигурацией  $k$ -предсказывающего алгоритма анализа будем подразумевать тройку  $(x, \alpha, \pi)$ , где

- $x \in \Sigma^*$  — непросмотренная часть входной цепочки, причём  $u \in \text{FIRST}_k^G(x)$ ;
- $u \in \Sigma^{*k}$  — 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. Определим следующую конфигурацию в зависимости от значения элемента управляющей таблицы  $M(X, u)$ :

1. Если  $M(X, u) = (B, i)$ , то  $(x, X\alpha, \pi) \vdash (x, \beta\alpha, \pi i)$ .
2. Если  $M(X, u) = \text{pop}$ , и в этом случае всегда  $X = a \in \Sigma$ ,  $x = ax'$ ,  $x' \in \Sigma^*$ , то  $(x, X\alpha, \pi) = (ax', a\alpha, \pi) \vdash (x', \alpha, \pi)$ .
3. Если  $M(X, u) = \text{accept}$ , что бывает только по достижении конечной конфигурации  $(\varepsilon, \$, \pi)$ , то анализатор останавливается, принимая входную цепочку.
4. Если  $M(X, u) = \text{error}$ , то анализатор сообщает об ошибке и останавливается, не принимая входную цепочку.

## 11. Построение 1-предсказывающего алгоритма анализа по LL(1)-грамматике и его обоснование. Алгоритм 2.1. Теорема 2.4

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

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

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

Положим  $\mathcal{A} = (\Sigma, \Gamma \cup \{ \$ \}, \Delta, M, X_0, \$)$ , где

- $\Sigma = V_T$ ;
- $\Delta = \{ 1, \dots, \#P \}$ ;
- $\Gamma = V_N \cup V_T$ ;
- $X_0 = S$ .

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

1.  $M(A, \alpha) = (\alpha, i)$ , если  $A \rightarrow \alpha$  является  $i$ -м правилом в  $P$  и  $a \in \text{FIRST}_1^G(\alpha)$ ,  $a \neq \varepsilon$ .

Если  $\varepsilon = \text{FIRST}_1^G(\alpha)$ , то  $M(A, b) = (\alpha, i)$  для всех  $b \in \text{FOLLOW}_1^G(A)$ .

2.  $M(a, a) = \text{pop} \quad \forall a \in \Sigma$ .
3.  $M(\$, \varepsilon) = \text{accept}$ .
4.  $M(X, a) = \text{error}$  для всех остальных  $(X, a)$ .

**Теорема 33.** Алгоритм производит правильный 1-предсказывающий алгоритм анализа для любой LL(1)-грамматики.

## 12. Определение операции $\oplus_k$ Лемма 2.1. Обоснование тождества $\text{FIRST}_k^G(\alpha\beta) = \text{FIRST}_k^G(\alpha) \oplus_k \text{FIRST}_k^G(\beta)$

**Определение 95.**  $\Sigma$  — алфавит,  $L_1, L_2 \subset \Sigma^*$ .

$$L_1 \oplus_k L_2 := \left\{ w \in \Sigma^{*k} \mid \begin{array}{l} x \in L_1, \quad y \in L_2, \\ \left[ \begin{array}{ll} w = xy, & |xy| \leq k, \\ xy = wz, & |w| = k \end{array} \right] \end{array} \right\}$$

**Лемма 16.** Для любой cfg  $G = (V_N, V_T, P, S)$  и любых цепочек  $\alpha, \beta \in V^*$  имеет место тождество

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

## 13. Анализ в LL( $k$ )-грамматиках. Определение 2.11. LL( $k$ )-таблицы. Алгоритм 2.2: построение множества LL( $k$ )-таблиц, необходимых и достаточных для анализа цепочек в данной LL( $k$ )-грамматике

**Определение 96.**  $G = (V_N, V_T, P, S)$  — cfg

Для каждого  $A \in V_N$  и  $L \subset V_T^{*k}$  определим  $T_{A \ L}$  — LL( $k$ )-таблицу, ассоциированную с  $A$  и  $L$  как функцию, которая по данной аванцепочке  $u \in V_T^{*k}$  выдаёт

- error, если не существует правила  $A \rightarrow \alpha$  такого, что  $u \in \text{FIRST}_k^G(\alpha) \oplus_k L$ ;
- $A$ -правило и конечный список подмножеств  $V_T^{*k}$ , если существует единственное такое правило;
- undefined, если подходящих правил несколько.

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

**Вход:**  $G = (V_N, V_T, P, S)$ .

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

1. Построить  $T_0 = T_{S \ \{\varepsilon\}}$  и  $\mathcal{T} = \{T_0\}$ .
2. Если  $T_{A \ L} \in \mathcal{T}$  и для некоторой цепочки  $u \in V_T^{*k}$  имеет место равенство  $T_{A \ L}(u) = (A \rightarrow x_0 B_1 x_1 B_2 \dots B_m x_m, \langle Y_1, \dots, Y_m \rangle)$ , то к множеству таблиц  $\mathcal{T}$  добавить таблицы из множества  $\{T_{B_i \ Y_i}\}_{i=1}^m$ .
3. Повторять шаг 2 до тех пор, пока ни одну новую таблицу не удастся добавить к  $\mathcal{T}$ .

## 14. Алгоритм 2.3: построение $k$ -предсказывающего алгоритма анализа. Пример 2.9. Теорема 2.5

**Алгоритм** (построение  $k$ -предсказывающего алгоритма анализа).

**Вход:**  $G = (V_N, V_T, P, S)$ .

**Выход:**  $\mathcal{A}$  — правильный предсказывающий алгоритм анализа для  $G$ .

1. Построим  $\mathcal{T}$  — множество необходимых  $LL(k)$ -таблиц для грамматики  $G$ .
2. Положим  $\mathcal{A} = (\Sigma, \Gamma \cup \{ \$ \}, \Delta, M, T_0, \$)$ , где  $\Sigma = V_T$ ,  $\Delta = \{ 1, \dots, \#P \}$ ,  $\Gamma = \mathcal{T} \cup V_T$ , где  $T_0 = T_{\$ \{ \varepsilon \}}$ .
3. Управляющую таблицу  $M$  определим на множестве  $(\Gamma \cup \{ \$ \}) \times \Sigma^{*k}$  следующим образом:
  - (а)  $M(T_{A \ L}, u) = (x_0 T_{B_1 \ Y_1} x_1 T_{B_2 \ Y_2} \dots T_{B_m \ Y_m} x_m, i)$ , если
 
$$T_{A \ L}(u) = (A \rightarrow x_0 B_1 x_1 B_2 \dots B_m x_m, \langle Y_1, \dots, Y_m \rangle),$$
 и  $A \rightarrow x_0 B_1 \dots B_m x_m$  является  $i$ -м правилом в  $P$ .
  - (б)  $M(a, av) = \text{pop} \quad \forall a \in \Sigma, v \in \Sigma^{*k}$ .
  - (в)  $M(\$ , \varepsilon) = \text{аксепт}$ .
  - (г)  $M(X, u) = \text{еррор}$  для всех остальных  $(X, u)$ .

**Теорема 34.** Алгоритм производит правильный  $k$ -предсказывающий алгоритм анализа для любой  $LL(k)$ -грамматики.

## 15. Тестирование $LL(k)$ -грамматик. Алгоритм 2.4. Определение 2.12

**Определение 97.**  $G = (V_N, V_T, P, S) - \text{cfg}$ ,  $A \in V_N$

$$\sigma(A) := \left\{ L \subset V_T^{*k} \mid \exists S \xrightarrow[\text{lm}]{*} w A \alpha, \quad w \in V_T^*, \quad L = \text{FIRST}_k^G(\alpha) \right\}$$

**Алгоритм (тестирование  $LL(k)$ -грамматик).**

**Вход:**  $G = (V_N, V_T, P, S) - \text{cfg}$ .

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

1. Для каждого нетерминала  $A$ , для которого существуют две или более альтернативы, вычисляется  $\sigma(A)$ .
2. Пусть  $A \rightarrow \beta$  и  $A \rightarrow \gamma$  — два различных  $A$ -правила. Для каждого  $L \in \sigma(A)$  вычисляется  $f(L) = (\text{FIRST}_k^G(\beta) \oplus_k L) \cap (\text{FIRST}_k^G(\gamma) \oplus_k L) = \emptyset$ .
  - Если  $f(L) \neq \emptyset$ , то алгоритм завершается с результатом ‘нет’.
  - Если  $f(L) = \emptyset$  для всех  $L \in \sigma(A)$ , то шаг 1 повторяется для других нетерминалов.
3. Повторить шаги 1 и 2 для всех нетерминалов.
4. Завершить алгоритм с результатом ‘да’.

## 16. Алгоритм 2.5: вычисление функции $\text{FIRST}_k^G(\beta)$ и её обоснование. Теорема 2.7

**Алгоритм (вычисление функции  $\text{FIRST}_k^G(\beta)$ ).**

**Вход:**  $G = (V_N, V_T, P, S) - \text{cfg}$ , и  $\beta = X_1 \dots X_n$ ,  $X_i \in V$ .

**Выход:**  $\text{FIRST}_k^G(\beta)$ .

Согласно лемме 16

$$\text{FIRST}_k^G(\beta) = \text{FIRST}_k^G(X_1) \oplus_k \dots \oplus_k \text{FIRST}_k^G(X_n),$$

так что задача сводится к вычислению  $\text{FIRST}_k^G(X)$  для  $X \in V$ .

Будем использовать метод последовательных приближений и строить последовательности множеств  $F_i(X)$  для всех  $X \in V_N \cup V_T$ .

$$1. F_i(a) = \{a\} \quad \forall a \in V_T.$$

$$2. F_0(A) = \left\{ x \in V_T^{*k} \mid \exists A \rightarrow x\alpha, \quad \left[ \begin{array}{l} |x| = k, \\ |x| < k, \end{array} \quad \alpha = \varepsilon \right] \right\}$$

3.

$$F_i(A) = F_{i-1}(A) \cup \left\{ x \in V_T^{*k} \mid \exists A \rightarrow Y_1 \dots Y_m, \quad x \in F_{i-1}(Y_1) \oplus_k \dots \oplus_k F_{i-1}(Y_m) \right\}$$

4. Так как  $F_{i-1}(A) \subset F_i(A) \subset V_T^{*k} \quad \forall A \in V_N$ , то шаг 3 требуется повторять, пока при некотором  $i = j$  не окажется  $F_j(A) = F_{j+1}(A) \quad \forall A \in V_N$ .

$$5. \text{FIRST}_k^G(A) := F_j(A).$$

### Теорема 35. $G - \text{cfg}$

Алгоритм правильно вычисляет функцию  $\text{FIRST}_k^G(\beta)$  для любой цепочки  $\beta$ .

## 17. Вычисление функции $\sigma(A)$ . Алгоритм 2.6. Пример 2.11.

**Алгоритм** (вычисление  $\sigma(A)$  для  $A \in V_N$ ).

**Вход:**  $G = (V_N, V_T, P, S) - \text{cfg}$ .

**Выход:**  $\sigma(A)$  для всех  $A \in V_N$ .

Введём в рассмотрение вспомогательную функцию

$$\sigma'(A, B) = \left\{ L \subset V_T^{*k} \mid \exists A \xrightarrow[\text{lm}]{*} wB\alpha, \quad w \in V_T^*, \quad L = \text{FIRST}_k^G(\alpha) \right\}$$

Очевидно, что  $\sigma(A) = \sigma'(S, A')$ . Будем вычислять  $\sigma'(A, B)$  методом последовательных приближений, параллельно выстраивая последовательности множеств  $\sigma'_i(A, B)$  для всех возможных пар  $(A, B) \in V_N^2$ .

$$1. \sigma'_0(A, B) = \left\{ L \subset V_T^{*k} \mid \exists A \rightarrow \beta B\alpha, \quad \beta, \alpha \in V^*, \quad L = \text{FIRST}_k^G(\alpha) \right\}$$

Пусть  $\sigma'_0(A, B), \dots, \sigma'_i(A, B)$  вычислены для всех пар нетерминалов.

2. Положим

$$\sigma'_{i+1}(A, B) = \sigma'_i(A, B) \cup \left\{ L \subset V_T^{*k} \mid \begin{array}{l} \exists A \rightarrow X_1 \dots X_m, \quad L' \in \sigma'_i(X_p, B), \quad X_p \in V_n, \\ 1 \leq p \leq m, \quad L = L' \oplus \text{FIRST}_k^G(X_{p+1} \dots X_m) \end{array} \right\}$$

3. Повторять шаг 2 до тех пор, пока не при некотором  $i = j$  не окажется, что  $\sigma'_{i+1}(A, B) = \sigma'_j(A, B)$  для всех  $(A, B)$ .

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

5.  $\sigma(A) = \sigma'(S, A)$ .

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

$$P = \{S \rightarrow AS, \quad S \rightarrow \varepsilon, \quad A \rightarrow aA, \quad A \rightarrow b\}$$

Вычислим функции  $\sigma(S)$  и  $\sigma(A)$  при  $k = 1$ . Сначала получаем

$$\text{FIRST}_1^G(S) = \{\varepsilon, a, b\}, \quad \text{FIRST}_1^G(A) = \{a, b\}$$

Затем строим последовательности  $\sigma'_i(X, Y)$ . За два шага получаем  $\sigma(S) = \{\{\varepsilon\}\}$ ,  $\sigma(A) = \{\{\varepsilon, a, b\}\}$ .

## 18. Алгоритм 2.7 вычисления функции $\text{FOLLOW}_k^G(A)$ . Теорема 2.9

**Алгоритм** (вычисление  $\text{FOLLOW}_k^G(A)$ ).

**Вход:**  $G = (V_N, V_T, P, S)$  — cfg.

**Выход:**  $\text{FOLLOW}_k^G(A)$  для всех  $A \in V_N$ .

Определим вспомогательную функцию

$$\varphi(A, B) = \left\{ w \in V_T^{*k} \mid \exists A \xrightarrow{*}_G \gamma B \alpha, \quad w \in \text{FIRST}_k^G(\alpha) \right\}$$

Очевидно, что  $\text{FOLLOW}_k^G(A) = \varphi(S, A)$ .

$$1. \varphi_0(A, B) = \left\{ w \in V_T^{*k} \mid \exists A \rightarrow \gamma B \alpha, \quad \alpha, \gamma \in V^*, \quad w = \text{FIRST}_k^G(\alpha) \right\}$$

2. Пусть значения  $\varphi_0(A, B), \dots, \varphi_i(A, B)$  уже построены для всех  $(A, B)$ .

$$\varphi_{i+1}(A, B) = \varphi_i(A, B) \cup \left\{ w \in V_T^{*k} \mid \begin{array}{l} \exists A \rightarrow X_1 \dots X_p X_{p+1} \dots X_m, \quad X_p \in V_N, \\ w \in \varphi_i(X_p, B) \oplus_k \text{FIRST}_k^G(X_{p+1}, \dots, X_m) \end{array} \right\}$$

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  $(A, B)$ . Тогда  $\varphi(A, B) = \varphi_j(A, B)$ .

4.  $\text{FOLLOW}_k^G(A) = \varphi(S, A)$  и  $\text{FOLLOW}_k^G(S) = \varphi(S, S) \cup \{ \varepsilon \}$ .

**Теорема 36.**  $G$  — cfg

Алгоритм правильно вычисляет функцию  $\text{FOLLOW}_k^G(A)$  для любого  $A \in V_N$ .

## 19. Теорема 2.8 — обоснование правильности вычисления функции $\sigma(A)$ для любого нетерминала $A \in V_N$ и $k \geq 0$

**Теорема 37.**  $G = (V_N, V_T, P, S)$  — cfg

Алгоритм правильно вычисляет функцию  $\sigma(A)$  для любого нетерминала  $A$ .

## 20. $k$ -предсказывающий алгоритм трансляции

**Алгоритм** (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  $\text{LL}(k)$ .

**Выход:**  $\mathfrak{Z}$  —  $k$ -предсказывающий алгоритм трансляции, реализующий трансляцию  $\tau(T)$ .

1. Предполагая, что множество  $\text{LL}(k)$ -таблиц, необходимых для анализа в грамматике  $G_i$  уже построено, положим

$$\mathfrak{Z} = (\Sigma, \Gamma \cup \{ \$ \}, \Delta, M, X_0, \$),$$

где  $\Sigma$  и  $\Delta$  такие же, как в схеме  $T$ ,

$$\Gamma = \mathcal{T} \cup \Sigma \cup \Delta', \quad \Delta' = \{ b' \mid b' = h(b), \quad b \in \Delta \}, \quad \Sigma \cap \Delta' = \emptyset, \quad X_0 = T_0 = T_S \{ \varepsilon \}$$

(а)  $M(T_{A \ L}, u) = x_0 y'_0 T_{A_1 \ L_1} \dots T_{A_m \ L_m} x_m y'_m$ , если

$$\begin{cases} T_{A \ L}(u) = (A \rightarrow x_0 A_1 x_1 \dots A_m x_m, \langle Y_1, \dots, Y_m \rangle), \\ A \rightarrow x_0 A_1 x_1 \dots A_m x_m, y_0 A_1 y_1 \dots A_m y_m - i\text{-е правило схемы} \end{cases}$$

Здесь  $y'_i = h(y_i)$ .

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

(с)  $M(b', u) = \text{pass}$  для всех  $u \in \Sigma^{*k}$ . Такой управляющий элемент определяет переход

$$(x, b' \alpha \$, y) \vdash (x, \alpha \$, yb)$$

(d)  $M(\$ , \varepsilon) = \text{ассерт.}$

(е)  $M(X, u) = \text{ерог}$  для всех остальных  $(X, u)$ .

**Теорема 38.**  $T = (N, \Sigma, \Delta, R, S)$  — простая семантически однозначная схема синтаксически- управляемой трансляции с входной грамматикой  $G_i$  класса  $\text{LL}(k)$  и  $\mathfrak{S} = (\Sigma, \Gamma \cup \{ \$ \}, \Delta, M, T_0, \$)$  —  $k$ -предсказывающий алгоритм трансляции, построенный посредством алгоритма.

Тогда  $\tau(T) = \tau(\mathfrak{S})$ .

**Теорема 39.**  $T$  — простая семантически однозначная схема синтаксически-управляемой трансляции с входной грамматикой  $G_i$  класса  $\text{LL}(k)$ .

Существует детерминированный магазинный преобразователь  $P$  такой, что

$$\tau_e = \{ (x \$, y) \mid (x, y) \in \tau(T) \}$$

## 21. Неразрешимые и разрешимые проблемы, касающиеся формальных языков

И этого нет...

## 22. Алгоритмически разрешимые проблемы, касающиеся конечных автоматов (проблемы пустоты и бесконечности языков, распознаваемых конечными автоматами, проблема эквивалентности конечных автоматов)

**Теорема 40.** Множество цепочек, принимаемых конечным автоматом с  $n$  состояниями,

1. непусто **тогда и только тогда**, когда он принимает цепочку длиной меньше  $n$ ;
2. бесконечно **тогда и только тогда**, когда он принимает цепочку длиной  $n \leq l \leq 2n$ .

**Следствие.** Существуют алгоритмы, разрешающие вопрос о пустоте, конечности и бесконечности языка, принимаемого любым данным конечным автоматом.

**Теорема 41.** Существует алгоритм для определения, являются ли два конечных автомата эквивалентными.