

ЛЕКЦИЯ 1

INTRODUCTION TO R o F L

Automata Specifications

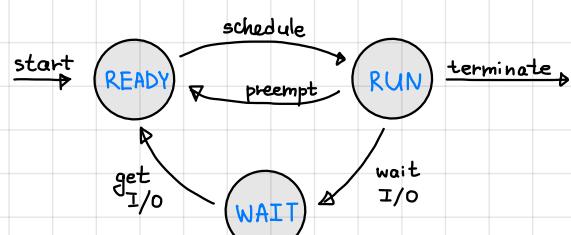
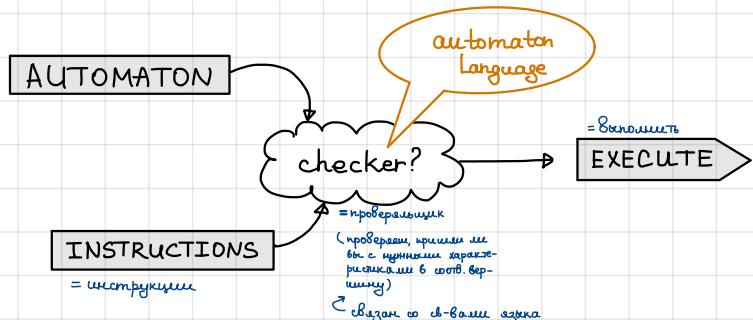


Схема на основе автоматов используется:

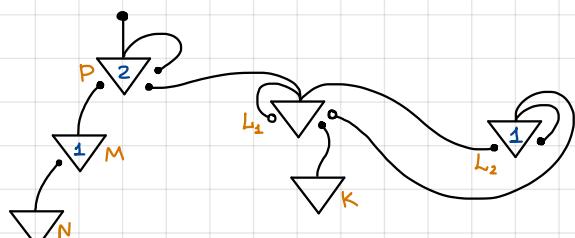
- communication protocols
- user interfaces
- блоки управления (control units)
- optimisation cycles (циклы опт.)

Programming: Automata Languages



- One-by-one input steps \Rightarrow достаточно модели автомата
- Sequence of multiple steps given simultaneously \Rightarrow требуется, чтобы executor (исполнитель) проверил пос-ть действует по модели автомата.
- Допустимые пос-ти шагов \Rightarrow формальный язык машин.

McCulloch & Pitts Neural Networks



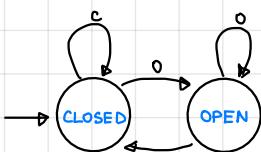
NN состоит из переходов, именующих либо маркер возбуждающего состояния, либо маркер затухающего сост. Помимо притока сколько-то возб. сигналов, чтобы нейрон активировался.

Клики (1951) представил регулярный язык, опис. событие в McCulloch & Pitts NN в терминах 3-х операций: {•, ∪, *}.

- – возбуждающий сигнал (excitatory signal)
- – затухающий сигнал (inhibitory signal)
- ∇ – входной нейрон
- ∇ – внутренний нейрон, срабатывающий каждый раз, когда не срабатывает ни один из затух. сигналов и по крайней мере к возбуждающим сигналам.

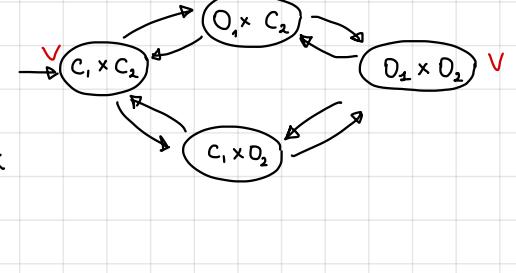
Automata models

Door automaton:



3 2 какана переходы

open	closed
O_1	C_1
O_2	C_2



$O_1 D_2 C_2 O_2 C_1 C_1$ – OK

$\times \quad O_1 D_2 C_2 C_2 C_1$ – не OK

$O_1 D_2 C_1 C_2$ – не OK

Formal Languages Formally (Формальный язык)

Definition: \mathcal{A} алгебру $\mathcal{A} = \langle M, \mathcal{F} \rangle$

Формальный язык – это набор термов M (конечных или ∞) над сигнатурой \mathcal{F} .

Сигнатура – это набор конструкторов (вместе с их арностью) и констант (необходимо) – ? итоговые термины в линейных констр.

Пример: 1) $\mathcal{A} = \langle \Sigma, \cdot \rangle$, где \cdot – операция конкатенации, и $M \subseteq \Sigma^*$, где $*$ – это операция

2) Tree Lang. automata, syntax trees, process graphs

1) Σ^n – мн-во членок длины n

2) Σ^+ – мн-во всех ненулевых членок Σ

$$\Sigma^+ = \Sigma^0 \cup \Sigma^4 \cup \Sigma^2 \cup \dots$$

Примеры формальных языков

- $\{ \underbrace{aa \dots}_{\text{1 поз}} \underbrace{abb \dots b}_{\text{1+3 поз}} \mid n \in \mathbb{N} \}$ — описание $\{ a^n b^{3n} \mid n \in \mathbb{N} \}$

3) Σ^* — мн-во всех цепочек над алф. Σ

$$\Sigma^* = \Sigma^+ \cup \{\epsilon\}$$

SYNTAX

- слова, содержащие квадратное кол-во букв "a"

$$\{ w \mid \exists k \left(\underbrace{|w|_a}_{\text{число букв "a" в слове } w} = k^2 \right) \}$$

("Я сплю, я сплю, я сплю")

SEMANTICS

- Все тautологии в классической логике
- Все постр-ко типизированные программы на Python
- формальные языки с linear-time-decidable membership (разрешимые во времени принадлежности)

Язык L является разрешимым, если $\exists f$ (f — ^{оруж.}бесконечная) т.е., если
 $w \in L \Rightarrow f(w) = 1$

$$w \in L \Rightarrow f(w) = 0$$

Язык L является нерешимым, если \exists бесконечное f , т.е.

$$w \in L \Rightarrow f(w) = 1$$

$$w \in L \Rightarrow f(w) = 0 \text{ или } f(w) \text{ — защищается}$$

Размывидности представлений FL

$$\{ w \mid w \text{ doesn't contain subword } ab \}$$

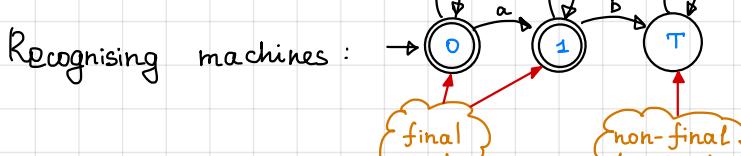
$$b^* a^*$$

пустое слово

Перенес термин: $\begin{cases} a \rightarrow aa \\ b \rightarrow bb \end{cases}, \{ \epsilon, a, b, ba \}$

P.S.: представление на основе автоматов легко сводится к машинам Тьюринга

↳ полезно при оценке вычислимости множества распознавания языка



$$\forall x, y \underbrace{(Q_a(x))}_{x=a} \& \underbrace{S(x,y)}_{y \text{ наследует } x} \Rightarrow \underbrace{\neg Q_b(y)}_{y \neq b}.$$

x, y — буквы

$S(x,y)$

$$Q_\gamma(x) \leftrightarrow (x = \gamma) \quad \xrightarrow{x \mid x}$$

Преобразование и анализ представлений

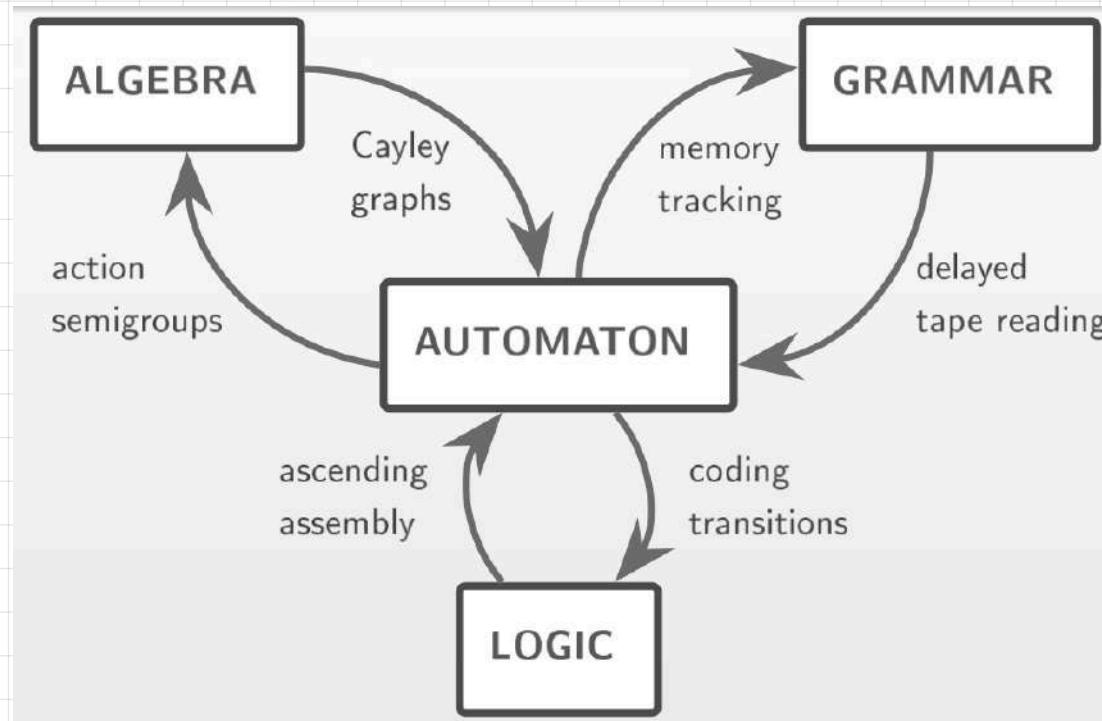
Переход от одного представления к другому — как и с тем что едят 😊

1) Как сделать корректный переход из одного класса представлений к другому классу?

2) Построение оптимальной формы представления внутри класса

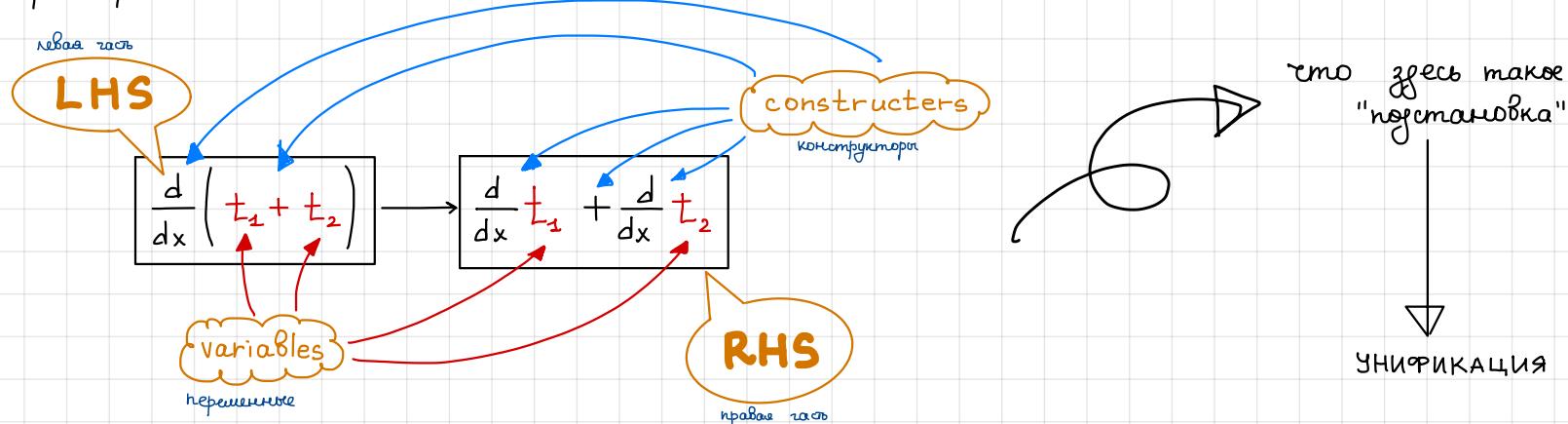
P.S.: темы больше представлений (содержанием разных), темы тем более нужно анализировать

— все представления одинаково устойчивы



Система переносования термов

* дифференцирование:



Унификация и сопоставление с образами

↓ Если у нас есть какое-то выражение, которое содержит переменные (в том числе и повторно вхождение)

и у нас есть правило переносования (Left & Right parts), то чтобы помочь, применяется ли правило

переносование к этому выражению целиком, надо нужно проверить:

- 1) Есть пара подстановок, которая одна применяется к термину, а вторая применяется к левой части прав. пере.
- 2) если все их применения и получилось одно и то же \Rightarrow правило переносования можно применять
→ надо нужно смотреть условие подстановки

Unification Formally (Унификация)

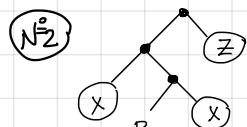
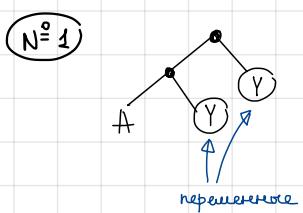
Definition: Унификацией термина t_1, t_2 унификатором называется пара подстановок

$$\langle \Theta_1, \Theta_2 \rangle, \text{т.к. } t_1\Theta_1 = t_2\Theta_2.$$

Правильное переносование обозначает термин t и LHS и правила $s_1 \rightarrow s_2$ путём построения замены

Пример: проанализировать 2 дерева, чтобы проверить, при каких условиях деревья будут и и равных

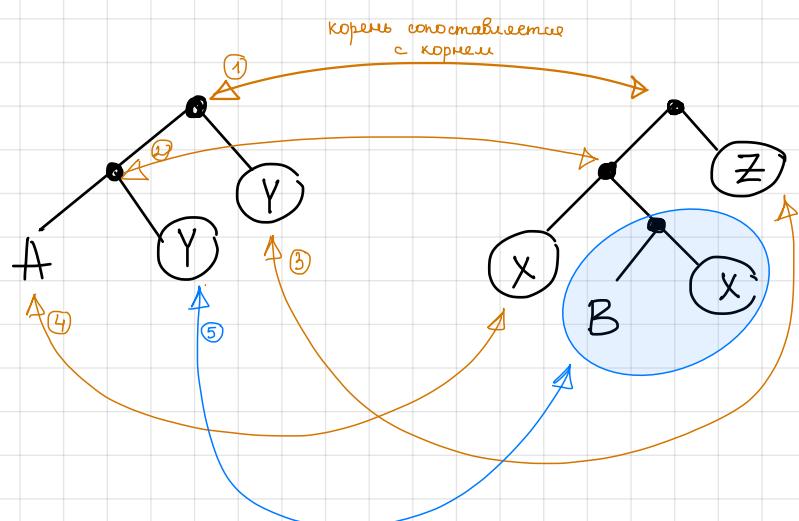
∃ деревесные переменные X, Y, Z.



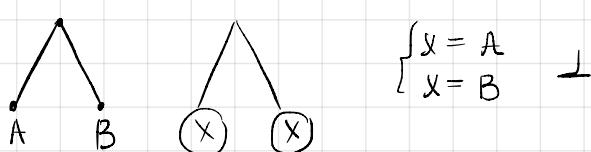
} два дерева

∃ м-во алгоритмов унификации

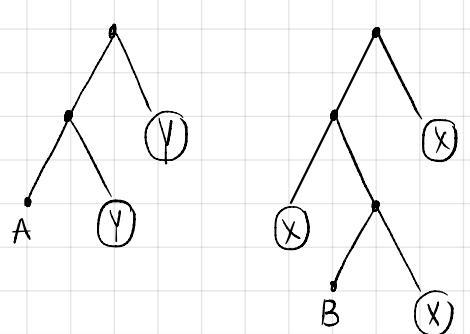
правый ↓ левый



- 1) корень соотв. с корнем
- 2) левые и правые ветвичатые дочерние соотв. друг с другом
- 3) $Y = Z$
- 4) $X = A$
- 5) $Y = X \Rightarrow$ решаем ит.



$$\left\{ \begin{array}{l} X = A \\ X = B \end{array} \right. \perp$$



невозможность ур-ки

$$\left\{ \begin{array}{l} X = A \\ Y \neq X \\ Y = B \end{array} \right. \text{унифицирование невозможно}$$

$$\left\{ \begin{array}{l} Y = A \\ Y = B \end{array} \right. \perp \text{невозм.}$$

Конструкции обозначений: постановки (substitutions)

Постановка в алгебре обозн. θ , её применение к эл-пу Φ обозн. через $\theta(\Phi)$.

Но в шам. языке расширено постфиксное обозначение $\Phi\theta$.

В классических учебниках substitution обозн. через $[x/A]$ \Rightarrow постфиксная запись: $P(x,x)[x/A]$.

Сейчас: $[x := A]$ и $[x \mapsto A]$

Редукция (reduction)

Дано TRS $\{\Phi_i \rightarrow \Psi_i\}_{i=1}^n$ и терм T

Основные правила унификации

- Redex это подтерм T_0 , который можно обозначить с каким-либо Φ_i с помощью унификации $\langle \Theta_1, \Theta_2 \rangle$
- Редукция заменение $T_0\Theta_1$ в T на $\Psi_i\Theta_2$
- Терм T — нормальная форма, если T не содержит больше redex
- Нормализация — приведение редукции в норм. формулу: $T \rightarrow T'$.

Пример: Дано TRS $\{0 + x \rightarrow x, 0 \cdot x \rightarrow 0\}$ и терм $(0 + 1) \cdot 0$ содержит redex $0 + 1$

вариант. $(0 + \underline{(0 + 0)}) + \underline{1}) \cdot (0 + 0)$

вариантов

редукция:

$$\frac{(0+1)}{1 \rightarrow 0} \cdot 0$$

$$0 \cdot 1 \longrightarrow 0$$

где явные правила неравн.: $x \cdot y = y \cdot x$

λ -преобразование и λ -эквивалентность

- если x, z — переменные, правило неравенства $\{eq(x,x) \rightarrow \text{True}\}$ и $\{eq(z,z) \rightarrow \text{True}\}$ — экв-ые
 $\{eq(x,z) \rightarrow \text{True}\}$ — не экв-ые
- λ -преобразование — это неравенство неравн. с сохр-ием семантики.
 Неприв-но в случае обл. перемен.

λ -исчисление

- Конструторы: применение (M,N) и $\lambda x.M$, оракул. $x \notin M$.
- Правило неравенства: применение $((\lambda x.M), N) \rightarrow M[x := N]$

P. S.: Reduction + capture-avoiding substitutions = универсальное ядро исчисления.

Proving properties of FL & Term Ordering

Check that the language $\mathcal{L} = \{w \mid |w|_a \text{ is even}\}$ is the set of normal forms generated from term srt $\{S\}$ using the following TRS T : $S \rightarrow a S a \quad S \rightarrow b S \quad S \rightarrow S b \quad S \rightarrow \varepsilon$

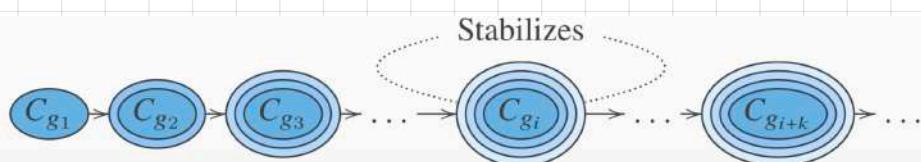
- Check that all words from \mathcal{L} can be generated by T .
- Check that all the normal forms η s.t $S \rightarrow \eta$ are in \mathcal{L} .

Assume that there exists w s.t. $S \rightarrow w$ and $|w|_a$ is odd. The reduction step before the application of $\{S \rightarrow \varepsilon\}$ can always be replaced by $\{S \rightarrow \varepsilon\} \Rightarrow$ making the word w shorter. We can repeat this process to ∞ , contradicting that $|w|$ is finite.

Key moment: the length ordering admits no infinite descending chains.

Ноэтериевое ядро вд-бо как упорядоченность (или нестрессость) или организованность.

Noetherian Orders & Well-Quasi-Orders



Ordering as an Induction Base

- A preorder \preceq is **well-founded** on set \mathcal{A} if it admits no infinite descending chains, i.e. all chains $t_0 \succeq t_1 \succeq \dots$ are finite.
- A preorder is **Noetherian** on set \mathcal{A} if it stabilizes upwards (no infinite strictly ascending chains, abbreviated ACC).
- A preorder \preceq is a **well-quasi-order** on set \mathcal{A} if every infinite term sequence from \mathcal{A} contains t_i, t_j s.t. $(i < j \wedge t_i \preceq t_j)$.

Well-founded Monotone Algebras

A preorder \preceq is monotone on \mathcal{A} , if

$$\forall f, t_1, \dots, t_n, s, s' \in \mathcal{A} \left(s \preceq s' \Rightarrow f(t_1, \dots, s, \dots, t_n) \preceq f(t_1, \dots, s', \dots, t_n) \right),$$

and is strictly monotone, if converse always fails.

Well-Founded Monotone Algebras

WFMA over the signature F on a well-founded set $\langle \mathcal{A}, > \rangle$ is an algebra s.t. for any $f \in F$ there exists a function $f_{\mathcal{A}} : \mathcal{A}^n \rightarrow \mathcal{A}$ being strictly monotone wrt all its arguments.

- $\langle \mathbb{N}, > \rangle$ can be a WFMA over a signature $(f_1, 2), (f_2, 1)$, given e.g. interpretations $f_1(x, y) = x + y + 1, f_2(x) = 2 \cdot x$;
- $\langle \mathbb{Q}^+, > \rangle$ and $\langle \mathbb{Z}, > \rangle$ are not WFMAs.

TRS Termination

Дано мн-во переменных \mathcal{V} , расширение $\delta : \mathcal{V} \rightarrow \mathcal{A}$ опред. как:

- $[x, \delta] = x\delta$
- $[f(t_1, \dots, t_n), \delta] = f_{\mathcal{A}}([t_1, \delta], \dots, [t_n, \delta]).$

TRS $\{l_i \rightarrow r_i\}$ совместим с WFMA $\mathcal{A} \Leftrightarrow$ для всех i, δ условие $[l_i, \delta] > [r_i, \delta]$ выполнено.

A TRS $\{l_i \rightarrow r_i\}$ terminates \Leftrightarrow there exists a WFMA compatible with $\{l_i \rightarrow r_i\}$.

The TRS $\left\{ \frac{d}{dx}(t_1 + t_2) \rightarrow \frac{d}{dx}t_1 + \frac{d}{dx}t_2 \right\}$ is terminating, which is verified by WFMA \mathcal{N} over $\langle \mathbb{N}, > \rangle$:

- $+_{\mathcal{N}}(u, v) = u + v + 1$;
- $\frac{d}{dx_{\mathcal{N}}}u = 2 \cdot u$.

Indeed,

$$\begin{aligned} \frac{d}{dx_{\mathcal{N}}}(t_1 + t_2) &= 2 \cdot (t_1 + t_2 + 1); \\ \frac{d}{dx_{\mathcal{N}}}t_1 + \frac{d}{dx_{\mathcal{N}}}t_2 &= 2 \cdot t_1 + 2 \cdot t_2 + 1. \end{aligned}$$

Примитивные:

$$\frac{d}{dx}(t_1 + t_2)$$

шаг-число t_1

2

↓

$$\left(\delta(t_1) + \delta(t_2) \right) + 1$$

2

↓

$$\frac{d}{dx}t_1 + \frac{d}{dx}t_2$$

↓

$$(2 \cdot \delta(t_1) + 2 \cdot \delta(t_2)) + 1$$

Это выраж. на единицу меньше, чем

⇒ при этом семантика, которая гарантируется, что это будет завершающее для t_1, t_2

Свободный monad

Usual case: the string data type is a carrier of a TRS; the only constructor is the string concatenation.

Then the term rewriting system becomes a *string rewriting system* (SRS): $\{l_i \rightarrow r_i\}$, where l_i, r_i are strings.

Терминальные & нетерминальные символы

Терминальные — это "конечные" эн-то апраксины, из которых строят строки языка.

Они не разбиваются далее по правилам грамматики и вы. составляют частичную языка, они же языки.

Нетерминальные — это вспомогательные символы, которые используют для описания структур языка

Они вводятся с помощью правила грамматики на основе терм. и/или других нетерм. символов.

Можно перенес. терм. и нетерм., но это \neq мн-во симв, порождаемых сист. перенес., которое входит в терминального апракс. Вот это язык

Formal Grammars (Грамматика)

Definition: Система правил переносов, кот. восп. из терм./нетерм. апракс.

or

Корректн. $G = \langle N, \Sigma, P, S \rangle$, где

- N — нетерм. апракс
- Σ — терм. апракс

- P - строка нерекурс. системы $\{\alpha_i \rightarrow \beta_i\}$, где $\alpha_i = \emptyset$
- $S \in N$ - начальный нетерм.

Language (язик $L(G)$)

Definition: Мн-бо слов, которые породяются в терм. апфавте из нач. символов
or $\{u \mid u \in \Sigma^* \text{ & } S \xrightarrow{*} u\}$, где $\xrightarrow{*}$ - композиция редукций.

P.S.: про завершаемость грамматики

$\times \quad S \xrightarrow{} SS$ — можно не завершаема
 $S \xrightarrow{} a$

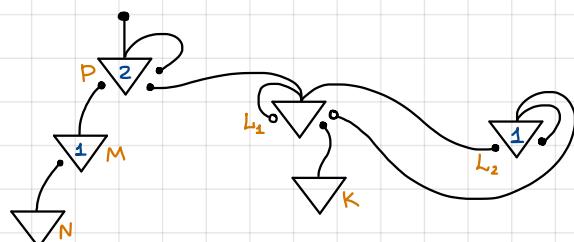
Перевернём! $\xrightarrow{2} \quad SS \xrightarrow{} S$ можно придумать инвер-цию, NFMA
 $a \xrightarrow{} S$ которые характеризуют, что не будет
 работы ∞

ЛЕКЦИЯ 2

REGULAR LANGUAGES

Напоминание из 1.1: пер. языки проходят от NN by McCulloch - Pitts

McCulloch & Pitts Neural Networks



- - возбуждающий сигнал (excitatory signal)
- - затухающий сигнал (inhibitory signal)
- ▽ - выходной нейрон
- ▽ - внутренний нейрон, срабатывающий всякий раз, когда не срабатывает ни один из затух. сигналов и по крайней мере, к возбуждающим сигналам.

\otimes, \neg - избыточные операции, относящиеся к языку, зад. операций $*$, \cdot , \vee .

Так появилось понятие регулярных языков.

Regular Expressions by Kleene

Definition-1: Пер. выраж. наз. амфавитом Σ опред. рекурсивно:

- 1) Если $\gamma \in \Sigma$ или $\gamma = \epsilon \rightarrow \gamma - \text{P.B.}$
- 2) Если $r - \text{P.B.}$, то $(r)^* \rightarrow \text{P.B.}$
- 3) Если $r_1, r_2 - \text{P.B.}$, то $r_1 r_2 - \text{P.B.}$ и $r_1 | r_2 - \text{P.B.}$

Definition-2: При зад. амфавите Σ P.B. явн. либо буква в Σ , ϵ или результат след. операций ($r_1, r_2 - \text{P.B.}$)

- $r_1 | r_2$ - обединение. $\mathcal{L}(r_1 | r_2) = \mathcal{L}(r_1) \cup \mathcal{L}(r_2)$
- $r_1 r_2$ - конкатенация. $\mathcal{L}(r_1 r_2) = \{w_1 w_2 \mid w_1 \in \mathcal{L}(r_1) \text{ и } w_2 \in \mathcal{L}(r_2)\}$
- $(r_1)^*$ - итерация (0 или более конкат. r_1 в единицах)

$$\mathcal{L}((r_1)^*) = \{\epsilon\} \bigcup_{i=1}^{\infty} \mathcal{L}(r_1)$$

↖ неподвласт. токса r_1^i

$$a^0 = \epsilon, \quad a^n, \quad a^* = \bigcup_{n \geq 0} a^n$$

r^+ - положит. итерация (это rr^*)

$r^?$ - выбор ($r|\epsilon$)

Приоритет: star > concatenation > union :

$$ab^* | c^*d \Leftrightarrow (a(b^*)) | ((c^*)d)$$

Academic regexes
• $, \cdot, ^*$ (sometimes $+, ?$) operations;
• define regular languages;
• studied in university courses (compilers & formal languages)
• Almost identical names are used for completely different (although related) notions.

REGEX (extended regexes)
• lookaheads, backreferences, etc;
• define non-context-free languages;
• used in practice (PCRE2 standard).

RE - академ. P.B.

REGEX - практическое P.B.

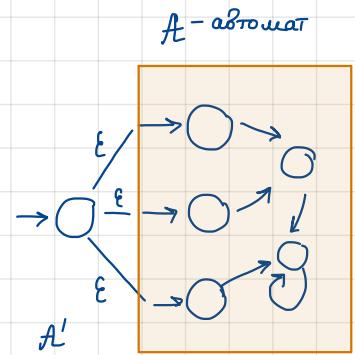
Недeterminированный конечный автомат (NFA)

Definition: $A = \langle \Sigma, Q, q_0, F, \delta \rangle$

- Σ - алфавит
- Q - множество состояний
- $q_0 \in Q$ - начальное состояние
- $F \subseteq Q$ - финальное состояние
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow 2^Q$ - правила перехода
также $\delta(q, a) = \{q'\}$ для NFA

Многа пишут: $\langle q_1, a, q_2 \rangle \in \delta \Leftrightarrow \langle q_1, a, M \rangle \in \delta \text{ & } q_2 \in M$

Или обычно: $q_1 \xrightarrow{a} q_2$

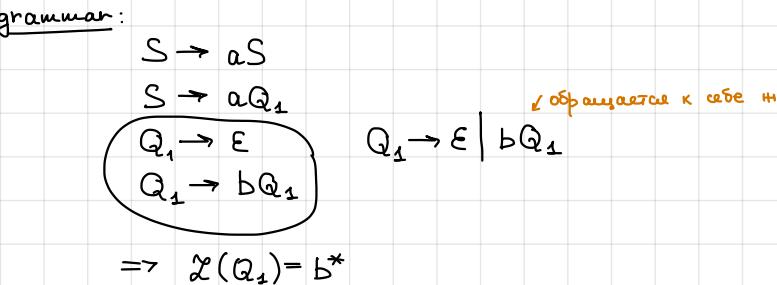
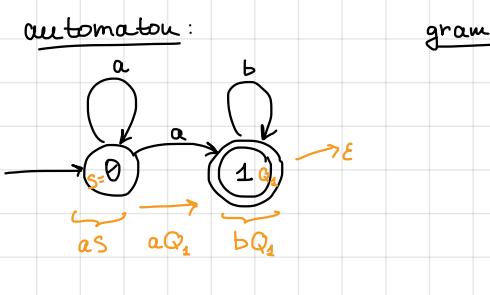


Классическое определение
Страна A , земя A !

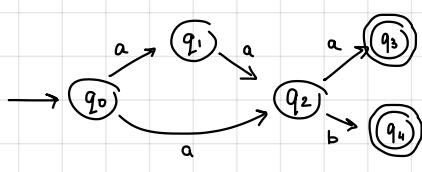
- Classical works (Kleene, Brzozowski): multiple NFA starting states are allowed.
- Modern formal language theory: the unique starting state in NFA is assumed.
- Equivalent (we can add an unique starting state with ϵ -transitions to the multiple states), but confusing (e.g. in Brzozowski minimisation).

Кодирование в грамматике

- 1) Состояние \Rightarrow нетерминал, $q_i \rightarrow S$ (\Rightarrow - преобразование)
- 2) $\langle q_i, a, q_j \rangle \Rightarrow Q_i \rightarrow aQ_j$
- 3) Если $q_i \in F$, то правило $Q_i \rightarrow \epsilon$



automaton:



grammar:

$$\begin{array}{ll} S \rightarrow a[q_1] & [q_2] \rightarrow a[q_3] \\ S \rightarrow a[q_2] & [q_2] \rightarrow b[q_4] \\ [q_1] \rightarrow a[q_1] & [q_3] \rightarrow \epsilon \\ [q_2] \rightarrow a[q_2] & [q_4] \rightarrow \epsilon \end{array}$$

Лемма Ардена

Если язык \mathcal{L} удовл. уравнению $\mathcal{L} = \mathcal{L}_1 \mathcal{L} \cup \mathcal{L}_2$, где $\epsilon \neq \mathcal{L}_1$, тогда $\mathcal{L} = \mathcal{L}_1^* \mathcal{L}_2$.

Доказ $X = \psi X \mid \psi$. Тогда $\mathcal{L}(X) = \psi^* \psi$.

Proof: Let us consider arbitrary $\omega \in \mathcal{L}$.

- If $\omega \in \mathcal{L}_2$, then the statement trivially holds.
- Otherwise, $\exists \omega_1 \in \mathcal{L}_1, \omega' \in \mathcal{L} (\omega = \omega_1 \omega')$. The suffix ω' also belongs to $\mathcal{L}_1 \mathcal{L} \cup \mathcal{L}_2$, and $|\omega'| < |\omega|$, since $\omega_1 \neq \epsilon$. Now we can repeat the same reasoning for ω' , and due to finiteness of $|\omega|$ and well-foundedness of $(\mathbb{N}, <)$ we will eventually get $\omega' \in \mathcal{L}_2$. \square

Arden's lemma allows one to solve the equation systems in Gaussian style, via non-terminal elimination + substitution, assuming there are no chain rules in the grammar.

Переходы по буквам в NFA (пока без ε-переходов)

1) Группируем правила перенесения по состояниям. Получаем вектора вида:

$$\{ Q_i = \alpha_1 Q_1 | \alpha_2 Q_2 | \dots | \gamma \}, \text{ где } \gamma - \text{терминал}$$

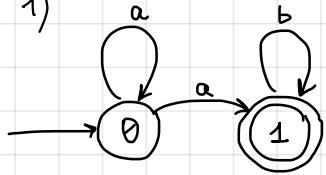
2) Выбрали Q_i через ин-бо всех состояний:

$$\{ Q_1, \dots, Q_{i-1}, Q_{i+1}, \dots, Q_n \} \text{ по цепи Аргена}$$

3) Добавляем, пока не выражим все

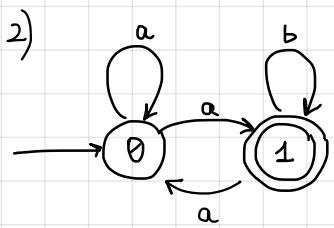
↪ порождающее регулярное выражение

Пример:



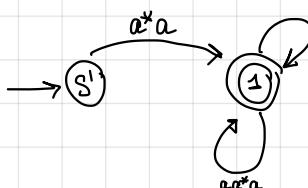
$$\begin{aligned} S &\rightarrow aS \mid aQ_1 \\ &\downarrow \text{н. Аргена} \\ S &\rightarrow a^* aQ_1 \\ Q_1 &\rightarrow bQ_1 \mid \epsilon \\ Q_1 &\rightarrow B^* \end{aligned}$$

$$\Rightarrow S \rightarrow a^* aB^*$$



$$\begin{aligned} S &\rightarrow aS \\ S &\rightarrow aQ_1 \\ Q_1 &\rightarrow \epsilon \\ Q_1 &\rightarrow bQ_1 \\ \vee Q_1 &\rightarrow aS \end{aligned}$$

это шаг из автомата
сделали фальшью.
теперь надо вернуть



$$\begin{aligned} S &\rightarrow aS \mid aQ_1 \\ S &\rightarrow a^* aQ_1 \\ Q_1 &\rightarrow bQ_1 \mid \epsilon \mid aS \\ Q_1 &\rightarrow (B|aa^*)Q_1 \mid \epsilon \\ Q_1 &\rightarrow (B|aa^*)^* \end{aligned}$$

ε-Closures and Chain Rules

- Any transition $q_i \xrightarrow{\epsilon} q_j$ corresponds to a chain rule $[q_i] \rightarrow [q_j]$ in the corresponding grammar G .
- state ϵ -closure is a closure set of the corresponding non-terminal N :
 $C(N) = \{N_i \mid \exists N'_1, \dots, N'_k (N \rightarrow N'_1 \wedge \dots \wedge N'_k \rightarrow N_i)\}$
I.e. $\langle N, N_i \rangle$ are pairs in a transitive closure \rightarrow_c^+ of the relation \rightarrow_c :
 $A_i \rightarrow_c A_j \Leftrightarrow (A_i \rightarrow A_j \in G)$.
- Before removing all chain rules, for every $N' \in C(N)$ and a non-chain rule $N' \rightarrow \Phi$, we add the transition $N \rightarrow \Phi$ to the set of grammar rules. Exactly as in the ϵ -closure algorithm for NFA.

Initial grammar:

$$\begin{array}{lll} S \rightarrow Q_1 & S \rightarrow Q_3 & Q_1 \rightarrow aQ_2 \\ Q_3 \rightarrow bQ_4 & Q_2 \rightarrow Q_5 & Q_4 \rightarrow Q_5 \\ & Q_5 \rightarrow \epsilon & \end{array}$$

After removing chain rules:

$$\begin{array}{ll} S \rightarrow aQ_2 & S \rightarrow bQ_4 \\ Q_2 \rightarrow \epsilon & Q_4 \rightarrow \epsilon \end{array}$$

Note: unreachable non-terminals Q_1, Q_3, Q_5 are deleted from the resulting grammar.

Метод устранения состояний (или метод решения уравнений)

1) Добавляем самое стартоное стат. "S" и финальное стат. "T"

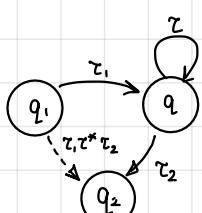
Добавляем ϵ -переход $S \xrightarrow{\epsilon} q_0$ и по всем $q \in F$ добавляем $q \xrightarrow{\epsilon} T$

Все состояния в Q считаются общими.

2)* Для состояния q строки цикла $q \rightarrow q$, но не имеющие пос-тью символов на пути, не проход. через другие стат.

Сообщ. цикл имеет вид τ .

Выход переходов:



Две всех переходов из q_1 в q_2 через q
строка переходов вида:

$$q_1 \xrightarrow{\tau_1, \tau_2^*, \tau_2} q_2$$

Удаляем q .

Недостаток метода: не ограничивается на ϵ -переходах

④ $NFA \rightarrow \text{regul. exp}$

Переход P.B. \rightarrow NFA

1) Операции: терм, |, *, . m.e. отходы от операций и \neq терм. P.B.

← метод Томпсона

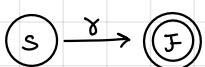
2) Буква — состояния

← переход Гиушкиова

Автомат Томпсона (рекурсивная сборка)

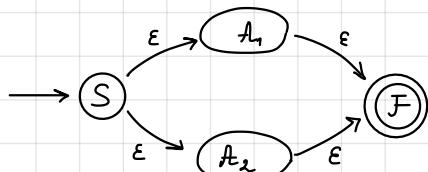
Каждое P.B. τ имеет рекурсивную структуру. Давайте используем эту структуру для построения NFA.

- $\tau = \gamma, \gamma \in \Sigma \Rightarrow \mathcal{A}(\tau)$:

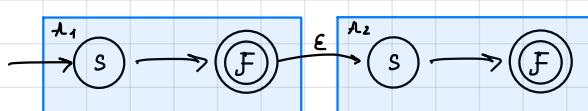


Any regular expression τ has a recursive structure. Let us use this structure to model the corresponding NFA.

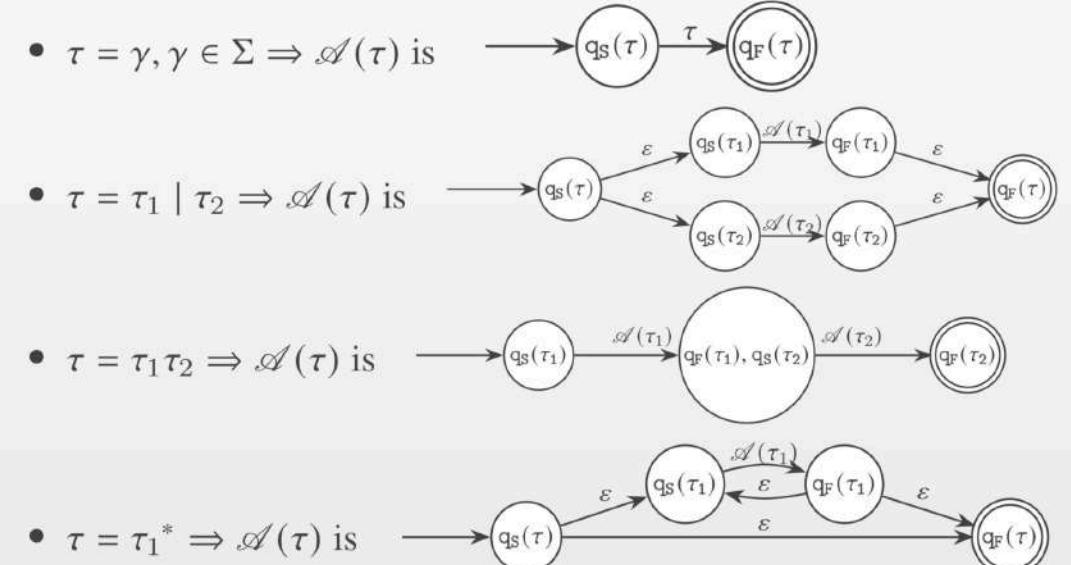
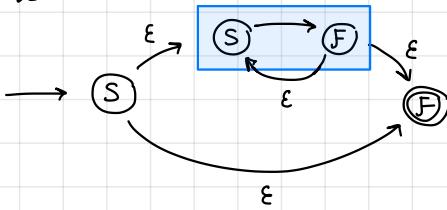
- $\tau = \tau_1 \cup \tau_2 \Rightarrow \mathcal{A}(\tau) = \mathcal{A}(\tau_1) \cup \mathcal{A}(\tau_2)$:



- $\tau = \tau_1 \tau_2$:



- $\tau = \tau^*$:



Свойства автомата Томпсона

- ① Двигается в структуру P.B.
- ② Единственное нач. сост., нет переходов в старт. и из начального

Автомат Гиушкиова

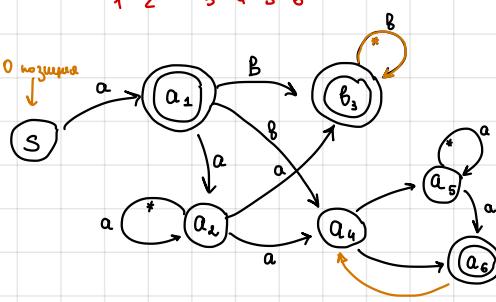
! устраивает ϵ -переходы, которые хар-кн для авт. Томпсона.

- 1) Вводим стартовое состояние S , соотв. термину "до начала строки"
- 2) Если же находимся в позиции к P.B. и имеем возможность перейти по буквам a, b сост. $i \rightarrow$ добавляем переход по a, b сост. a^i .
- 3) Если a, b сост. i' может быть продолжкой последней \rightarrow делаем сост. (a^i) промежуточными.

m.e. строим матрицу переходов.

Идея: P.B. — все буквы помеченных позициями

Пример: $a\alpha^*(b|\alpha^*\alpha)^*$

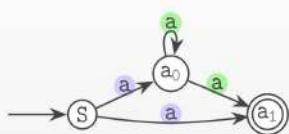


- на 1 сост. боязнь, что букв.
- недетерминизм такой же, как и в P.B.
- недетрм. сохраняется, а не \Rightarrow как в ав. Томпсона

ПРО ϵ -ПЕРЕХОДЫ И ϵ -ЗАМЫКАНИЯ

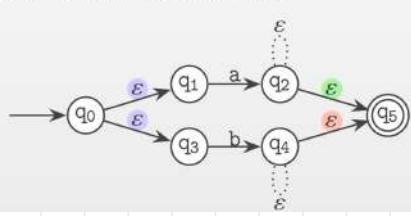
Sources of Non-determinism in NFA

- Transition sets wrt (with respect to) a letter $\gamma \in \Sigma$ that are not singletons.



т.е. ϵ -переход, это переход из q_i в q_j без записи символа из строки

- ϵ -transitions (so-called silent actions).



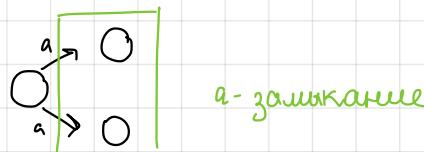
ϵ -closure (ϵ -закрытие)

ϵ -закрытие состояний: q — мн-во сост., достижимых из q по ϵ -переходам (но без a)

Зависка q на его ϵ -закрытие — это устранение ϵ -переходов из автомата A

Теорема

Альфред Гуцкович и Р.В. r = автомат Томсона после перехода к ϵ -закрытию.



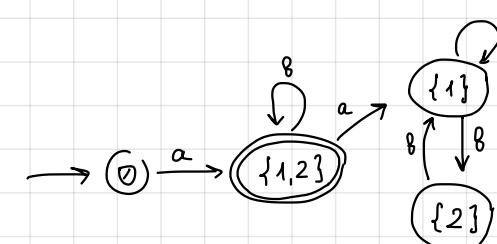
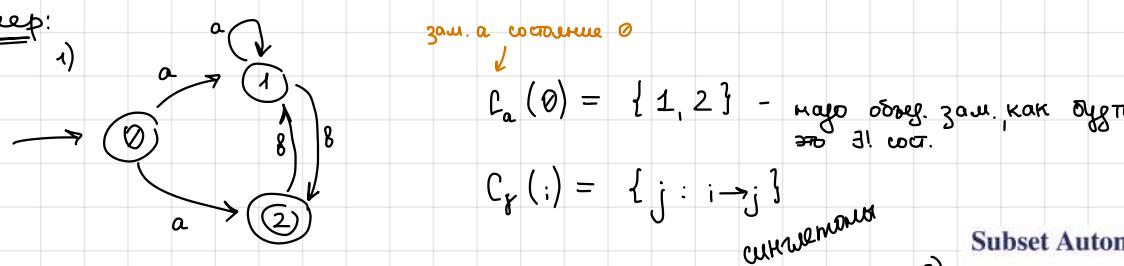
Закрытие состояния q по γ — мн-во $\{q_i \mid q \xrightarrow{\gamma} q_i\}$ (если $\gamma = \epsilon$, то считаем $q \xrightarrow{\epsilon} q$)

Закрытие всех сост. по всем символам Σ — демаршигация.

ϵ -Closures and Chain Rules

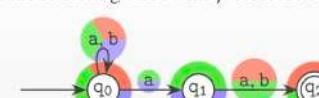
- Any transition $q_i \xrightarrow{\epsilon} q_j$ corresponds to a chain rule $[q_i] \rightarrow [q_j]$ in the corresponding grammar G .
 - state ϵ -closure is a closure set of the corresponding non-terminal N :
- $$C(N) = \left\{ N_i \mid \exists N'_1, \dots, N'_k (N \rightarrow N'_1 \& \dots \& N'_k \rightarrow N_i) \right\}$$
- I.e. $\langle N, N_i \rangle$ are pairs in a transitive closure \rightarrow_c^+ of the relation \rightarrow_c :
 $A_i \rightarrow_c A_j \Leftrightarrow (A_i \rightarrow A_j \in G)$.
- Before removing all chain rules, for every $N' \in C(N)$ and a non-chain rule $N' \rightarrow \Phi$, we add the transition $N \rightarrow \Phi$ to the set of grammar rules. Exactly as in the ϵ -closure algorithm for NFA.

Пример:

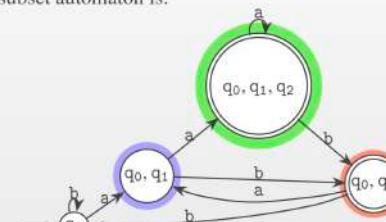


Subset Automaton: a Simple Example

Let us consider the following NFA with γ -closures of its states:



Hence, its subset automaton is:



Subset Automaton: a Simple Example

Let us consider the following NFA with γ -closures of its states:



- a-closure of starting state $\{q_0\}$ is $\{q_0, q_1\}$.
- b-closure of starting state $\{q_0\}$ is the state $\{q_0\}$ itself.
- a-closure of the state $\{q_0, q_1\}$ is $\{q_0, q_1, q_2\}$.
- b-closure of the state $\{q_0, q_1\}$ is $\{q_0, q_2\}$.
- a-closure of the state $\{q_0, q_1, q_2\}$ is $\{q_0, q_1, q_2\}$ itself, while b-closure is the state $\{q_0, q_2\}$.
- a-closure of the state $\{q_0, q_2\}$ is $\{q_0, q_1\}$, while b-closure is $\{q_0\}$.

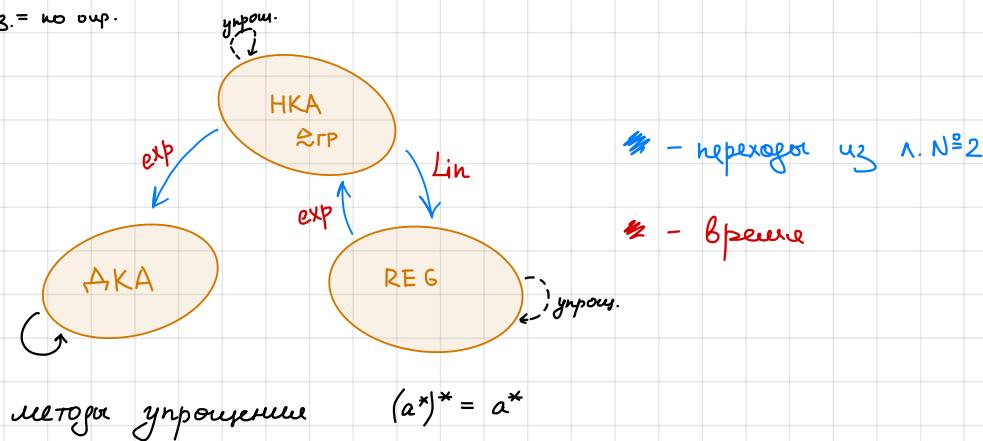
Алгоритм демаршигации (NFA \rightarrow DFA)

- Устранение ϵ -переходов по замыканию
- Демаршигизация по подм-зам.

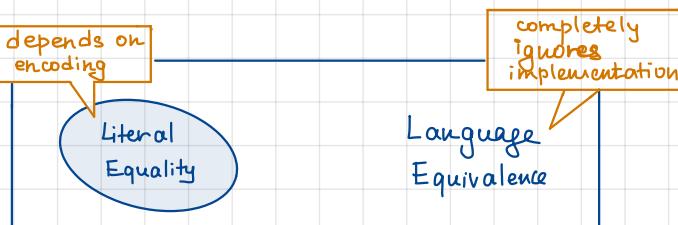
ЛЕКЦИЯ 3

EQUivalence OF FINITE AUTOMATA

\approx - приблиз.=но бир.

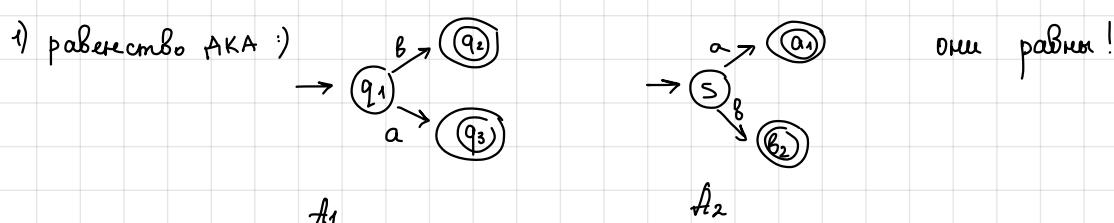


Machine comparison



Отношение эквивалентности

① Равенство (Вы прекрасно с регулярами)



Каноническое представление состояния и условия равенства ДКА

- 2) равенство НКА :
- надо сравнивать НКА по изображению
 - нужно про равенство
 - можно кратко:
 - ! - есть идея из НКА \rightarrow ДКА, но даже разных НКА может быть один и тот же ДКА

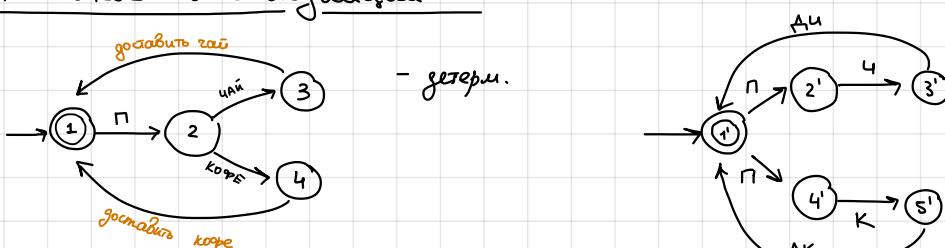


у них разные св-ва

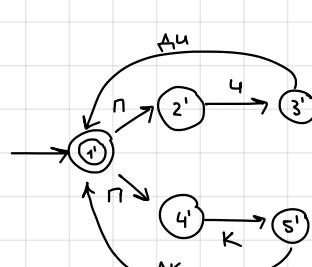
② Равенство представителей (Бисимметрия)

③ Равенство языков — эквивалентность (DFA - a-a-a-a)

Что такое "бисимметрическое"?



$\Pi(4\Delta 4 | KAK)^*$ — язык



A_i — г-т, что не фин.

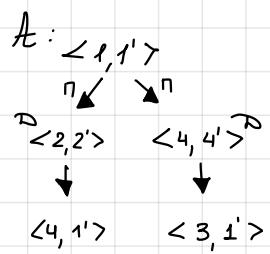
D_i — г-т, что финков

- ① А_i Выбираем A_i-автомат и делает переход по нек. символу γ $q_{N_i} \xrightarrow{\gamma} q_{(N+1)}$

$\leftarrow q_{N_1}, q_{N_2} \rightarrow$
A₁ A₂

β_i — это 2-й автомат
(так сокращение)

- 2) Δ б т_{3-i}, переходы по γ $q_{N(i-1)} \rightarrow q_{(N+1)i-1}$
 Δ побеждает, если Δ не может уничтожить ход
 Δ побеждает, если игра утилизируется вничью :)



О отношение Бисимуляции

LTS
 когда как автомата
 могут быть со
 оправдывая обеих

Пусть даны системы различающихся переходов T_1 и T_2 , в которых есть набор состояний

$$T_1 = \langle Q_1, \Sigma, \delta_1 \rangle, \Sigma - \text{автомат}$$

\uparrow \uparrow
 стартовый тупик. перехода

Излага отношение бисимуляции (обозн: \sim) ставят в соответствие $q_{i1} \sim q_{i2}$, если:

$$\forall \gamma \in \Sigma \left(\exists (q_{i1} \xrightarrow{\gamma} q_{k1}) \Rightarrow \exists (q_{i2} \xrightarrow{\gamma} q_{k2}) \right) \wedge \left(\exists (q_{i2} \xrightarrow{\gamma} q_{k2}) \Rightarrow (\exists (q_{i1} \xrightarrow{\gamma} q_{k1}) \wedge q_{i2} \sim q_{k2}) \right)$$

Bisimulation is a relation \sim between states of the systems T_1 and T_2 satisfying the following property:

- If $q_1 \sim q_2$ ($q_1 \in T_1, q_2 \in T_2$), then for every transition $q_1 \xrightarrow{\gamma} q'_1$ in T_1 there is a transition $q_2 \xrightarrow{\gamma} q'_2$ in T_2 such that $q'_1 \sim q'_2$, and vice versa.

Starting and final (if any) states must be bisimilar.

Every state machine \mathcal{A} can be represented as a labelled transition system.

- \mathcal{A}_1 and \mathcal{A}_2 are bisimilar \Leftrightarrow their LTS T_1 and T_2 are bisimilar.

КОМЕНДА. СОСТ. НЕ УЧИТАВАЮТСЯ

Labelled transition systems are not necessarily finite; moreover, LTS contain no final states.

Existence of final states can be modelled via introducing endmarkers (usually denoted $\$$). Then every final state of an NFA has a transition by the endmarker to the unique «bottom» state.

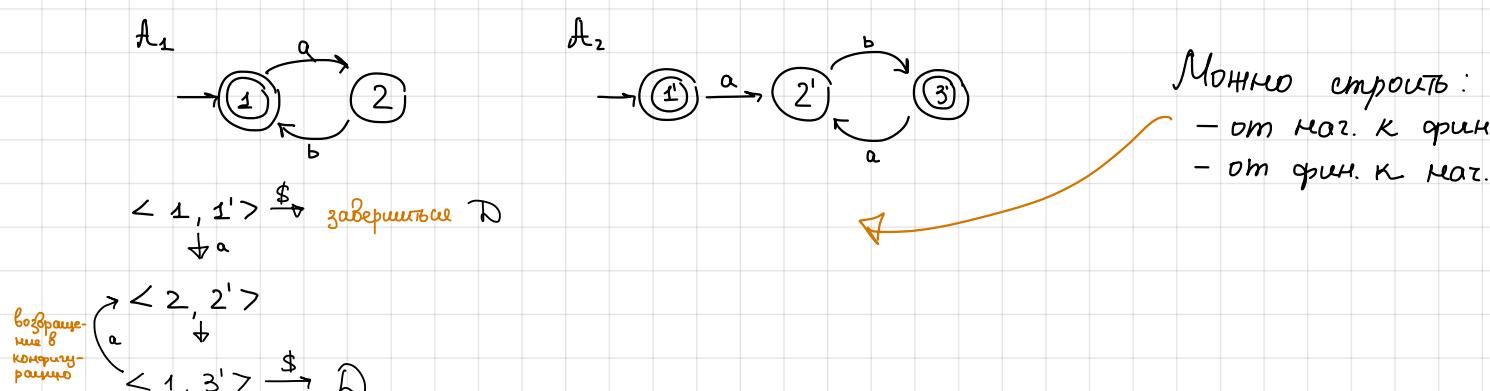
Если T_1 и T_2 – автомата (и.е. Эквивалент. сост.), то конечные сост.

должны быть бисимуляционно конечны:

$$\forall q_{i1}, q_{i2} (q_{i1} \in F_1 \wedge q_{i2} \sim q_{i1} \Rightarrow q_{i2} \in F_2) \text{ и обратно}$$

Equivalent trim DFA are bisimilar

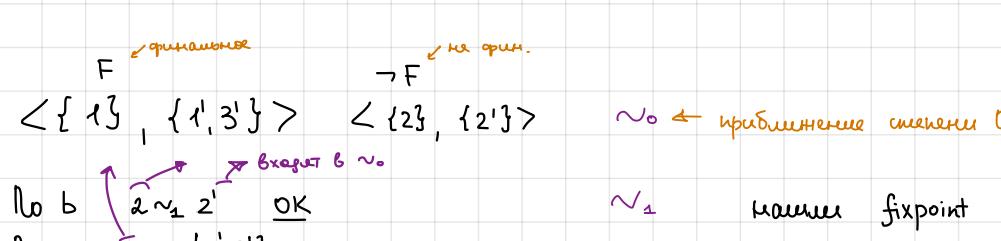
Даны 2 автомата



$$1) q_i \sim_j q_j, \text{ если } (q_i \in F) \Leftrightarrow (q_j \in F)$$

$$2) q_i \sim_m q_j, \text{ если } \forall \gamma \in \Sigma ((q_i \xrightarrow{\gamma} q_j) \wedge (q_j \xrightarrow{\gamma} q_i) \wedge (q_i \sim_n q_j)) \wedge (q_i \sim_m q_j)$$

3) О отношение \sim fixpoint (2)



k-Bisimulation: Playing Backwards

Given an NFA \mathcal{A} , how do we know that its states q_i, q_j are bisimilar?

- if q_i is final, while q_j is non-final, then \mathcal{A} can choose q_i and declare game-over, winning the game. Hence, \mathcal{A} can win doing no move at all. If q_i, q_j are both final or both non-final, at least one move is required for \mathcal{A} to win, and we say that $q_i \sim_k q_j$.
- if exists γ and q'_i s.t. $q_i \xrightarrow{\gamma} q'_i$, and for all q'_j s.t. $q_j \xrightarrow{\gamma} q'_j$ $q'_i \neq q'_j$, then \mathcal{A} wins in $k+1$ moves starting from the position (q_i, q_j) . Otherwise, we say that $q_i \sim_{k+1} q_j$.

A closer look on the \sim_{k+1} -condition:

$$q_i \sim_{k+1} q_j \Leftrightarrow \forall q'_i, \gamma(q_i \xrightarrow{\gamma} q'_i \Rightarrow \exists q'_j (q_j \xrightarrow{\gamma} q'_j) \wedge \forall q'_j, \gamma(q_j \xrightarrow{\gamma} q'_j \Rightarrow \exists q'_i (q_i \xrightarrow{\gamma} q'_i)))$$

When we reach the fixpoint of \sim_k (i.e. $\sim_k = \sim_{k+1}$), then we know that \mathcal{A} never can win in position (q_i, q_j) given $q_i \sim_k q_j$, hence $q_i \sim_{k+1} q_j$.

Fixpoint (неподвижные точки)

Definition: гара функции $f: D \rightarrow D$, её неподвижной точкой наз. значение $z \in D : f(z) = z$

If $f: \mathbb{R} \rightarrow \mathbb{R}$, then its fixpoints are located at the intersection of f -graph with the diagonal.

We have met the fixpoint construction before in the Arden lemma.
Namely, the expression $\Phi^*\Psi$ is the fixpoint of the function
 $f(X) = \Phi X \mid \Psi$.

Moreover, the «Kleene star» construction is a fixpoint itself: Φ^* is the fixpoint language for the function $s(X) = \Phi X \mid \varepsilon$.

The fixpoint construction is particularly useful in computer science when some set is saturated wrt a well-founded relation.

Бисимметричность в АКА и минимизация

Если в контоле классе экв-ти по бисимметрии ровно 1 эл-м \Leftrightarrow А-минимален (A - это АКА)
или

Дан АКА с набором состоян. Q и набором конечн. состоян. F , знаем, что $q_i \sim q_j \Leftrightarrow$

наборы распознанных симв., нач. с q_i и нач. с q_j , равны. След-ко, можно минимизировать АКА, обозначив бисимметричное состояние.

\Rightarrow : вся возможное мн-во \sum^* разбивается на А-ие подмн-ва префиксов:

$$(w_i \equiv w_j) \Leftrightarrow \exists q_i : (q_0 \xrightarrow{w_i} q_i \text{ & } q_0 \xrightarrow{w_j} q_j)$$

Если $\exists A'$ (число состоян. $A' <$ число состоян. A), тогда $\delta \equiv_{A'} \delta$ меньше к.э.

Возьмём w_1 и w_2 : $w_1 \not\equiv_A w_2$, $w_1 \equiv_{A'} w_2$

$$A : \begin{cases} q_0 \xrightarrow{w_1} q_i \\ q_0 \xrightarrow{w_2} q_j \end{cases} \quad q_i \not\sim q_j \Leftrightarrow \exists u (w_1 u \in L(A) \text{ & } w_2 u \notin L(A) \vee (w_2 u \in L(A) \text{ & } w_1 u \in L(A)))$$

но $q_k \xrightarrow{u} \text{ единст.} \Rightarrow L(A') \neq L(A)$

Соответствие супфиксов и минимальности DFA

Пусть дан регулярный язык L .

Считаем, что $w_1 \equiv_L w_2 \Leftrightarrow \forall u \in \sum^* ((w_1 u \in L) \Leftrightarrow (w_2 u \in L))$

отношение Майкиона-Корреа

Теорема Майкиона-Корреа

L регулярный \Leftrightarrow число классов экв-ти конечно.

□ :

\Rightarrow Т.к. $\exists w_1, w_2$ - слова из L
таких ч.з.

$L \neq \emptyset \Rightarrow \exists \text{dfa } A, \text{ т.к. } L = L(A)$
тогда $\{w_i\}$ может быть разбита на конечное
число ч.з. по \equiv_L
 \Rightarrow если w_i, w_j находятся в классе q_k , то
 $\forall u (w_i u \in L(A) \Leftrightarrow w_j u \in L(A))$ (1)

\Leftrightarrow Т.к. имеется конечное $\{w_i\} \equiv_L$

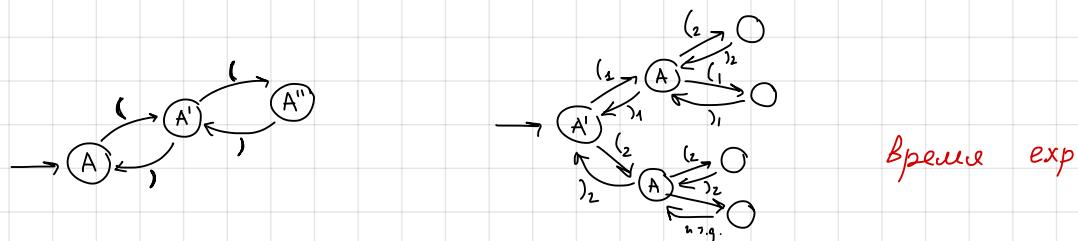
(1) Сартовым состоян. $\{w_i\}$ все ост. ч.з. - также состоят A

(2) $(w_i, q_i \equiv_L w_j) \Rightarrow \{w_i\} \xrightarrow{q_i} \{w_j\}$

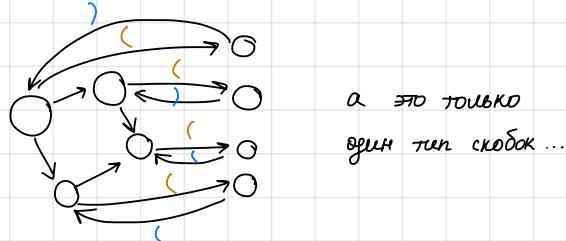
(3) Рассматриваемые состояния, т.к. $\{w_i\} \equiv_L$, то
 $\forall w_i \in \{w_i\} (w_i \in L)$...

Преим и угар со скобочками :)

→ без учёта вырожденности скобок

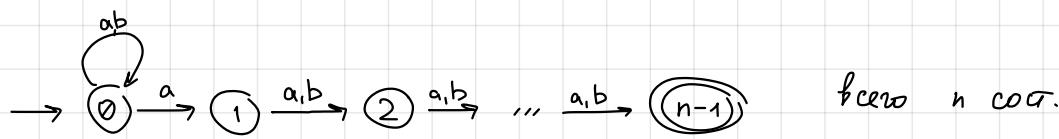


У нас есть "(" дис состояний, делаются двойные состояния для ")"



Почему exp- переход между НКА и АКА

$$(a|b)^* a (a|b)^n \text{ и авт. Гицкюба}$$



✗ на классах экв-ти по М-Н.:

нр п.	не короткое!		
	$a b^n$	b^n	b^{n-1}
1	+	-	-
2	+	+	-
3	+	-	+

⇒ Все префиксы вида $(a|b)^n$ формируют unique K.Э.

ЛЕКЦИЯ 4

$$\Sigma = \sum_{\text{алгоритм}} + \sum_{\text{внутренне}} + \sum_{\text{calls}} + \sum_{\text{return}}$$

алгоритм
 внутренне
 простое действие
 |
 calls
 |
 алгор. вызовов
 |
 return
 |
 алгор. возврата

VPA - Visibly Pushdown Automata

Без сост.

$$\{ Q_0, F, Q, \Sigma, \Gamma, \delta \}$$

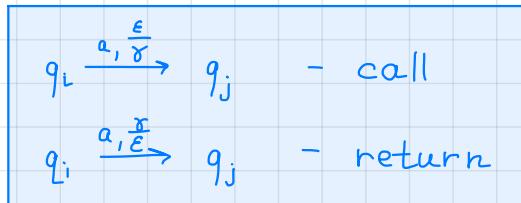
ин-бо нач. сост.
 |
 \$Q_0 \subseteq Q\$
 \$F \subseteq Q\$
 |

правила перехода

Если $a \in \Sigma_I$, то $\langle q_i, a, q_j \rangle \in \delta$ (иначе $q_i \xrightarrow{a} q_j$) - переход как в обычных автоматах

Если $a \in \Sigma_C$, то $\langle q_i, a \times \gamma, q_j \rangle \in \delta$ (иначе $q_i \xrightarrow{a/\gamma} q_j$) м.е. кладём γ в стек

Если $a \in \Sigma_R$, то $\langle q_i, a \times \gamma, q_j \rangle \in \delta$ (иначе $q_i \xrightarrow{a/\gamma} q_j$)



Конфигурации VPA: $\{ \langle q, \Phi_\perp^*, w_1 \dots w_n \rangle \}$

Φ_\perp^*
 остаток строки
 ↑
 ⊥-bottom
 (дно стека)

Если w_i - внутрн. $\Rightarrow \{ \langle q', \Phi_\perp^* \rangle, w_{i+1} \dots w_n \}$

$$q \xrightarrow{w_i} q'$$

Если w_i - call $\Rightarrow \{ \langle q', \gamma \Phi_\perp^* \rangle, w_{i+1} \dots w_n \}$

$$q \xrightarrow{w_i/\gamma} q'$$

Если w_i - return $\Rightarrow \Phi = \gamma \Phi' \& \{ \langle q', \Phi'_\perp^* \rangle, w_{i+1} \dots w_n \}$

$$q \xrightarrow{w_i/\gamma} q'$$

Достигши дна $q_i \xrightarrow{a/\perp} q_j : \{ \langle q_i, \perp, a w \rangle \} \Rightarrow \{ \langle q_j, \perp, w \rangle \}$

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

$L = \{ \text{ин-бо правильных скобочных последовательностей (ПСП) + баланс. внутрн. кот. строк из } \{a, b\} \}$

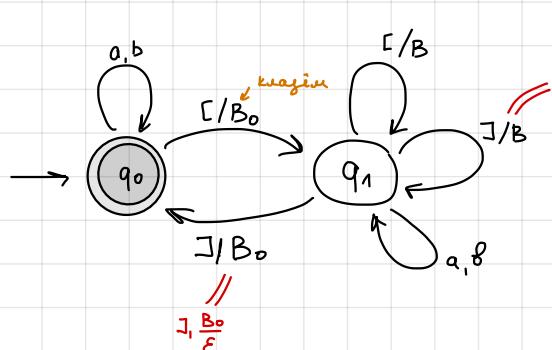
$$\Sigma_C = \{ [\] \}$$

исчезающие скобки

$$\Sigma = \{ [] \}$$

$$\Gamma = \{ B_0, B \}$$

все остальные



В q_1 стек имеет вид:

$$B^* B_0 \perp$$

м.е. на вершине встёгли метку B , метку B_0 , что означает баланс - то

✗ $a[8[]a]$:

$$\{ \langle q_0, \perp, a[8[]a] \rangle \}$$

$$\{ \langle q_0, \perp, [8[]a] \rangle \}$$

$$\{ \langle q_1, B_0 \perp, 8[]a \rangle \}$$

$$\{ \langle q_1, B_0 \perp, [3a] \rangle \}$$

$$\{ \langle q, BB_0 \perp,]a] \rangle \}$$

без метки B т.к.]

$$\{ \langle q_1, B_0 \perp, a \rangle \}$$

$$\{ \langle q_1, B_0 \perp,] \rangle \}$$

$$\{ \langle q_0, \perp, \epsilon \rangle \}$$

→ P

Definition:

VPL (Visibly Pushdown language) — такой язык над Σ , что:

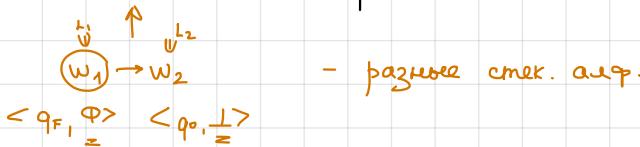
- \exists разбиение $\Sigma = \Sigma_I \cup \Sigma_C \cup \Sigma_R$
- \exists VPA A , such. $L(A)$ при таком разбиении

Тогда A_1, A_2 — VPA

1) $L(A_1) \cup L(A_2)$ — pacn. VPA (стек. символы у каждого могут быть свои)



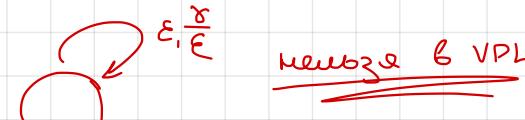
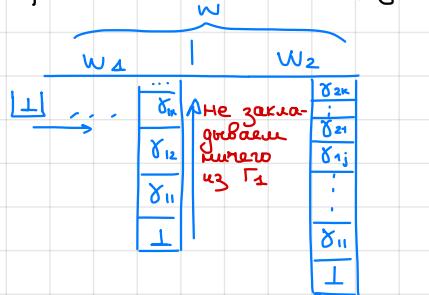
2) $L(A_1) \cap L(A_2)$ — pacn. VPA



2a) Γ_1 и Γ_2 делаем различными.

Все переходы вида $q_i \xrightarrow{a/\perp} q_j$ в A_2 дополняем переходами вида $q_i \xrightarrow{a/\gamma} q_j$, где $\gamma \in \Gamma_1$

2b) Применение конструкции Томпсона



иначе в VPL

иначе со стека нечего снимать

$\gamma_{11} \notin \Gamma_2$

3) $L(A_1) \cap L(A_2)$ — pacn. VPA A_3

Составление A_3 — пары $\langle q_{i1}, q_{i2} \rangle$, где $q_{i1} \in Q_1$, $q_{i2} \in Q_2$

Составление стека t_3 : строки вида $(\langle \gamma_{i1}, \gamma_{i2} \rangle)^* \langle \perp, \perp \rangle$, где $\gamma_{i1} \in \Gamma_1$, $\gamma_{i2} \in \Gamma_2$

- Для всех переходов вида $q_i \xrightarrow{a} q_j$ $q'_i \xrightarrow{a} q'_j$ (\leftarrow внутр. перех.), то добавляем $\langle q_i, q'_i \rangle \xrightarrow{a} \langle q_j, q'_j \rangle$

- Для всех $a \in \Sigma_C$ но переходами $q_i \xrightarrow{a/\gamma_i} q_j$ $q'_i \xrightarrow{a/\gamma_i} q'_j \Rightarrow \langle q_i, q'_i \rangle \xrightarrow{a/\langle \gamma_i, \gamma_i \rangle} \langle q_j, q'_j \rangle$

- Для всех $a \in \Sigma_R$ но переходами $q_i \xrightarrow{a/\gamma_i} q_j$ $q'_i \xrightarrow{a/\gamma_i} q'_j \Rightarrow \langle q_i, q'_i \rangle \xrightarrow{a/\langle \gamma_i, \gamma_i \rangle} \langle q_j, q'_j \rangle$

$\begin{vmatrix} \langle \gamma, \gamma \rangle \\ \langle \perp, \perp \rangle \end{vmatrix}$!!! совместимость по альфавитам

4) Если A — VPA, то дополнение $L(A)$ — VPL

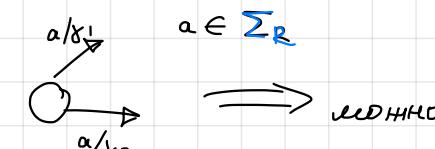
Definition

Дополнение языка — это все слова, которое представляется в том же самом альфавите, но не входит в наш язык.

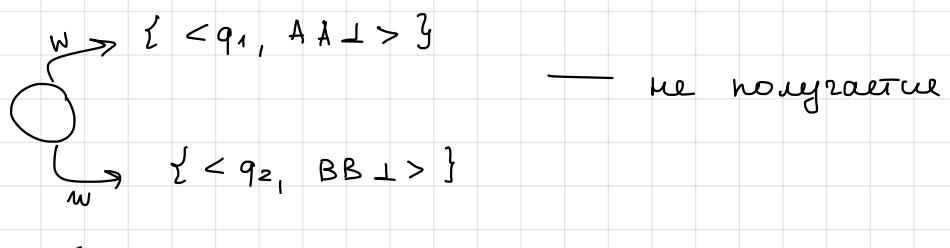
Детерминизацию VPA?

VPA детерм. если: 1) $\forall q_i$ все q_i есть максимум по 1 переходу по символам из $\Sigma_I \cup \Sigma_C$

2) $\forall q_i$ все q_i есть max по переходам по $\langle a, \gamma \rangle$, $a \in \Sigma_R$, $\gamma \in \Gamma$



как делать детерм-цию?



Если w — строка со сбачансир. вводом и возвратом, тогда

$$\langle q_i, \varphi \rangle \xrightarrow{w} \langle q_j, \varphi \rangle$$

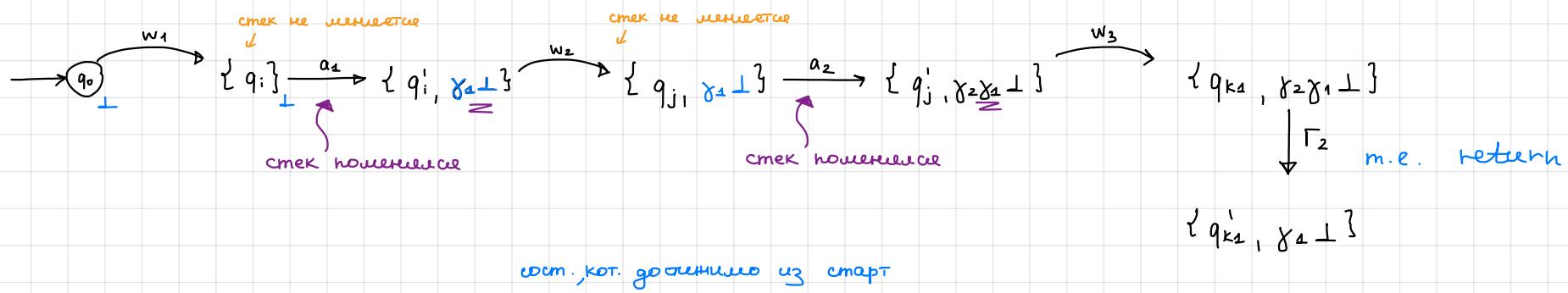
т.е. стек не меняется от изменения w

$$h: \Sigma_c \rightarrow [$$

$$\Sigma_r \rightarrow]$$

$$h(w) = \text{ЛСЛ откес. } [,]$$

$\nexists w_1 a_1 w_2 a_2 w_3 \Gamma_2$ w_1, w_2, w_3 — сбачансир. вводы ($\text{не осудь ванко, то там бокс}$)



Составление A^{DET} это пара (S, R)

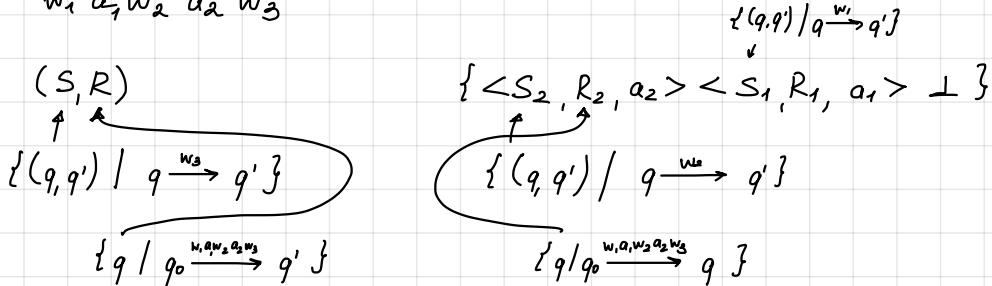
возможные переходы по сбачансир. симву

символе стека (S_i, R_i, a_i)

ши-бо пар сост., т.е. $q \xrightarrow{w_i} q'$ (q, q'), где w_i — кратчайшее сбачансир. слово от начального блокрина $a_i \in \Sigma_c$ сбачансир. слова

ши-бо сост., достичмо из старт. но предиксу до a_i

$\nexists w_1 a_1 w_2 a_2 w_3$



$$\Sigma = \Sigma_{\text{int}} \cup \Sigma_{\text{call}} \cup \Sigma_{\text{ret}}$$

(про VPA)

w -сбаланс. $\Leftrightarrow w \in \Sigma_{\text{int}}^*$ (с точки зрения действие на стек)

Что значит "сбалансированное"? Отобразим все символы

$$h \left(\begin{array}{l} \gamma \in \Sigma_{\text{call}} \quad \beta \quad (\\ \gamma \in \Sigma_{\text{ret}} \quad \beta \quad) \\ \gamma \in \Sigma_{\text{int}} \quad \beta \quad \epsilon \end{array} \right) \rightarrow \text{тогда } w\text{-сбаланс.} \Leftrightarrow h(w) \in \text{ПСП}$$

return
call

$\{\} \} - \text{сбаланс.}$

$h(\{\}) = (())$

Теорема Майхина-Нероде для VPL

1) Определение ии-ва Well-Matched, или $WM(\hat{\Sigma})$ — сбаланс. слова

Matched-Calls, или $MC(\hat{\Sigma})$: $h(w)$ — суффиксы ПСП

Matched-Return, или $MR(\hat{\Sigma})$: $h(w)$ — префиксы ПСП

2) Конгруэнции $w_1 \equiv_{WM} w_2 \Leftrightarrow \forall u, v \in \Sigma^* (uw_1v \in L \Leftrightarrow uw_2v \in L)$

и.е. если штасим эти слова (w_1, w_2) в середине строки, то они действуют одинаково с точки зрения ϵ -ти длины L .

Конгруэнции $u_1 \sim_0 u_2 \Leftrightarrow \forall v \in \Sigma^* (u_1v \in L \Leftrightarrow u_2v \in L)$

$u_1, u_2 \in MC$

но суфф. MC

Конгруэнции $u_1 \equiv_{MR} u_2 \Leftrightarrow \forall v \in MR (u_1v \in L \Leftrightarrow u_2v \in L)$

u_1, u_2

отнесены к кот. присваивается суфф. из MR

Язык L — VPL относ. $\hat{\Sigma}$ $\Leftrightarrow \equiv_{WM}, \sim_0, \equiv_{MR}$ конечные (и.е. имеют конечн. число классов экв-ти).

В иниции приводится док-во теоремы

$\nexists \{a^n b^n \mid n \in \mathbb{N}\}$. Докажем, что L — WM-VPL

$\Sigma_{\text{call}} = \{a\}$	\equiv_{WM}	$\langle \epsilon, \epsilon \rangle$
	$a^n b^n \rightarrow \epsilon$	+
$\Sigma_{\text{ret}} = \{b\}$	рабаб	-

все слова не соотв. рег. $a^* b^*$

$\nexists \{ww^k \mid w \in \{a, b\}^*\}$

Без отображения общности примем, что



$$\Sigma_{\text{call}} \cap \Sigma_{\text{ret}} = \emptyset$$

Попытуемся вписать $\Sigma_{\text{int}} = \{a, b\} - ?$

a^8	$+ \quad + \quad -$
$a^2 b^2$	$- \quad + \quad -$
\vdots	$\ddots \quad +$
$a^k b^k$	$- \quad - \quad +$
	∞
	\Downarrow
	не VPL

Рассмотрим выше две таких рассуждения:

Как доказать, что $L(\Sigma)$ — не VPL?

- 1) Построить возможное разбиение Σ на Σ^{int} , Σ^{call} , Σ^{return}
 - 2) Для каждого из разбиений Σ^k строим со серией Well-Matched слов, в которых различие классов якобы.
- ! Помимо про разбиение $\Sigma^{\text{int}} = \Sigma$ ($\Sigma^{\text{call}} = \Sigma^{\text{return}} = \emptyset$)

$\left. \begin{array}{c} < \quad > \\ < / \dots > \\ <, /, \dots > - \text{ в кавычках} \\ a^n b a^n \end{array} \right\}$ — не VPL

смогу ли я VPL без лексики

PDA — Pushdown Automaton или

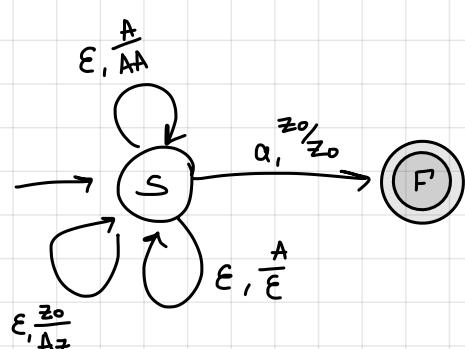
составляющие:

- Σ — символьные строки, алфавит
- Q — конечные состояния ($q_0 \in Q$)
- $F \subseteq Q$ — множество принятых состояний
- Γ — стек.алфавит
- δ — правила перехода

"стековый автомат"
"автомат с магазинной памятью"

$$\delta: Q \times \Gamma \times (\Sigma \cup \{\epsilon\}) \rightarrow Q \times \Gamma^*$$

- либо снимаем со стека символ
- либо снимаем и кладём др. символы
- либо не снимаем и кладём серию символов



символ, кот. мы видим на вершине стека
то, что мы закладываем на стек

z_0 — это стек, кот. с самого начала лежит в стеке

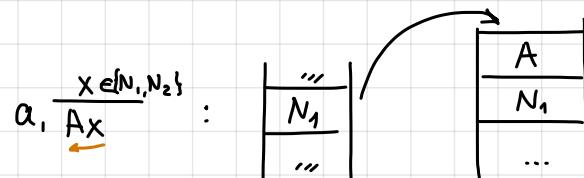
магаз. конориз.: $\langle q_0, z_0 \rangle$

A
A
...
A
z_0

$a, \frac{x}{Ax}$ — если на стеке "x", а хочу пополнить это-то, "a", то введение "Ax"

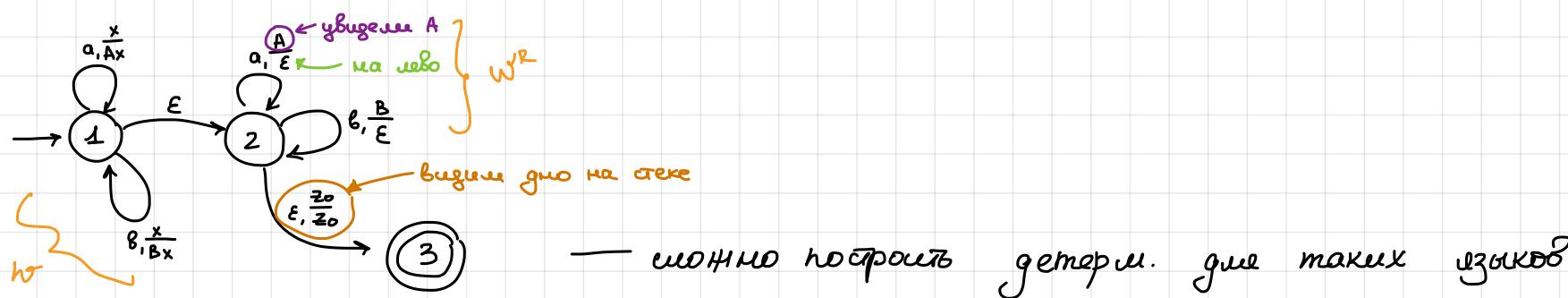
$a, \frac{x \in \{N_1, N_2\}}{Ax}$ — не ограничение

$a, \frac{x}{x}$



Пример:

1) Чёт. палиндромы $\{ww^R \mid w \in \{a,b\}^*\}$

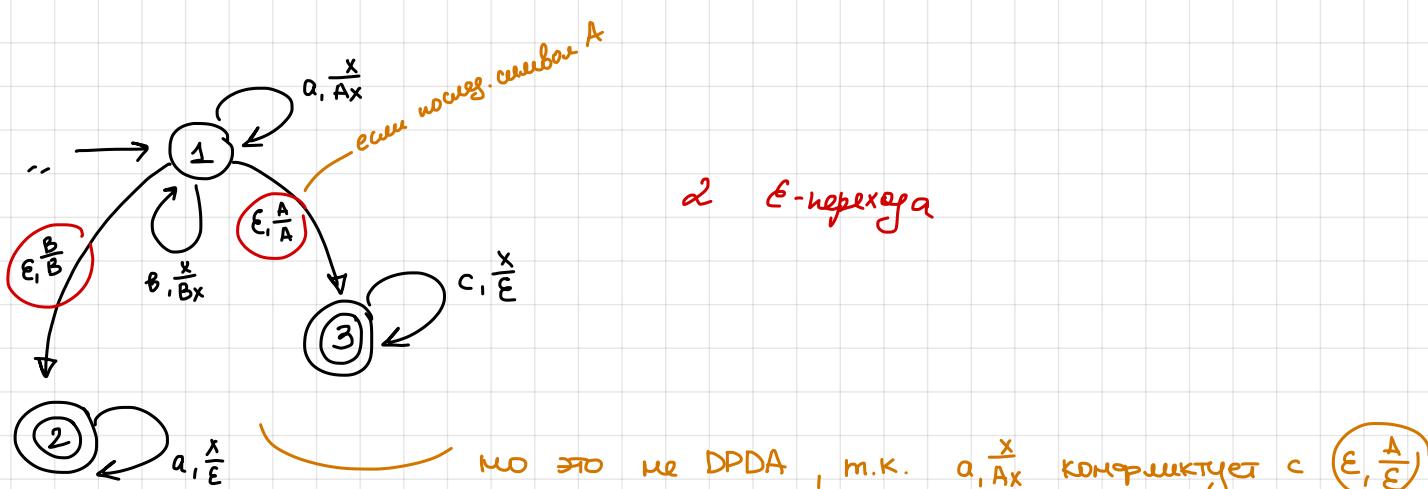


Детерминированный PDA (DPDA)

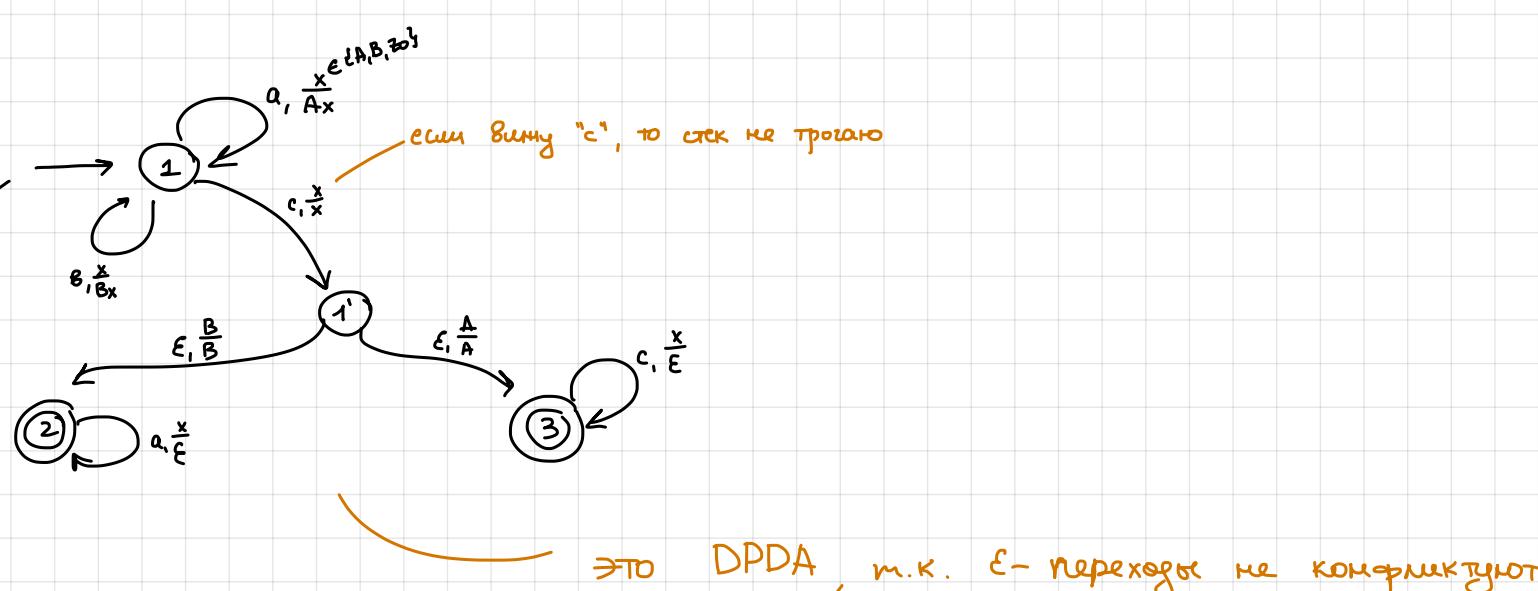
Ма б машина работает ограничения:

- 1) Если есть ϵ -переход $\langle q_i, A \rangle \xrightarrow{\epsilon} \langle q_j, \Phi \rangle$ но Вершина А, то нет других переходов по Вершине стека А.
- 2) По ^{всем} каждой из пар $\langle a_i, A_j \rangle$, $a_i \in \Sigma$, $A_j \in \Gamma$, существует максимум 1 переход.

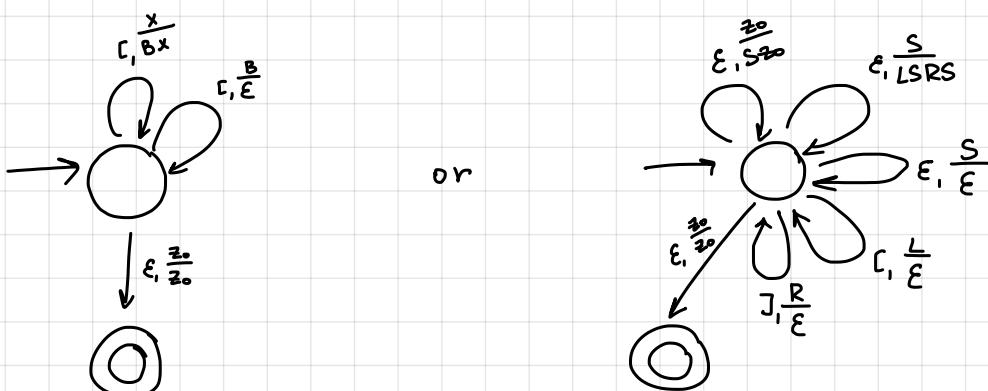
2)



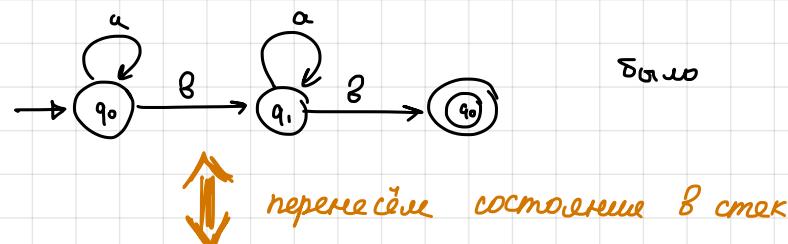
3)



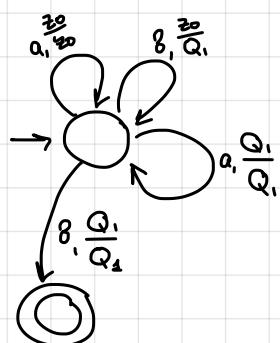
4) ПСР $\{[]\}$



5) $\hat{a}ba^*b$

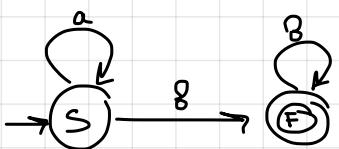


база

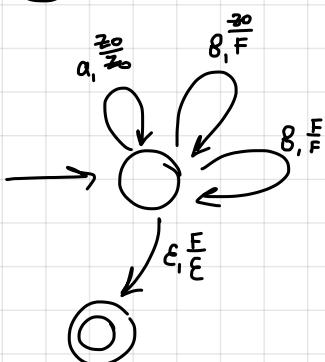


ЛЕКЦИЯ 6

CONTINUE ABOUT PDA
&
CONTEXT-FREE GRAMMAR (CFG)

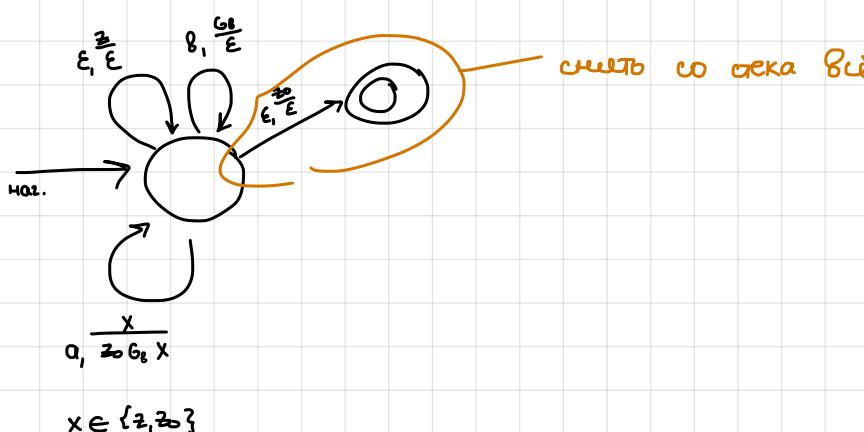


$S \rightarrow aS$
 $S \rightarrow bF$
 $F \rightarrow bF \mid \epsilon$



$S \rightarrow aSbS \mid \epsilon \Rightarrow S \rightarrow aSG_bS \mid \epsilon$
 $G_b \rightarrow z$

В стек кладём
нетерминалы,
что делать с "b".



Перевод от грамматики с правилами вида $A_i \rightarrow \varphi \left(\begin{smallmatrix} A; \in N \\ \varphi \in (N \cup \Sigma)^* \end{smallmatrix} \right)$ к PDA:

Σ - множество терм.
 N - множество нетерм.

- Сделаем переход из стартового состояния: $\langle q_0, z_0 \rangle \rightarrow \langle q_B, S z_0 \rangle$ старт. нетерм.
- По каждому правилу вида $A_i \rightarrow \varphi$, где $\varphi \in N^*$ строим переход $\langle q_B, A_i \rangle \xrightarrow{\varphi} \langle q_B, \varphi \rangle$

| Если в правило $A_i \rightarrow \varphi$ есть терминалы в правой части \Rightarrow
обращаем их в одиночные нетермы.

- $(A_j \in \Sigma)$ добавляем $G_j \rightarrow j$ и меняем j на G_j во всех прав. частях)

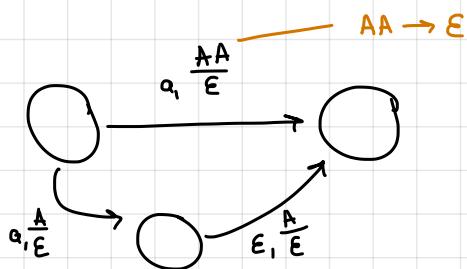
$A \rightarrow aA^*A$ (нетерм. в конце строки после то)

\uparrow
 G_B т.е. запоминаем значение, чтобы хранить его в стеке

- Добавляем по каждому правилу $A_i \rightarrow j$ ($j \in \Sigma$) строим переход $\langle q_B, A_i \rangle \xrightarrow{j} \langle q_B, \epsilon \rangle$
- Добавляем $\langle q_B, z_0 \rangle \xrightarrow{\epsilon} \langle q_F, \epsilon \rangle$ ($q_F \in F$)

Грамматики вида $A_i \rightarrow \varphi \left(\begin{smallmatrix} A; \in N \\ \varphi \in (N \cup \Sigma)^* \end{smallmatrix} \right)$ наз. КОНТЕКСТНО-СВОБОДНЫМИ.

[на вершине стека стоит
тот нетерминал, кот. должн
быть переписан]



нока умелец

KC → PDA

Теорема

KC-языки это такие же что языки, распознаваемые PDA

семи原因之一ные языки

→ aS_1BS_2a — строка из терм. и нетерм.

CFG — KC-грамматика
CFL — KC-язык

P.S. Охотим наше нормальное гр./яз.



KC-языки замкнуты относительно:

- 1) конкатенации, обединения, пересечения (см. конструкции для НКА)
- 2) пересечения с регулярным языком

Дадут A_1 -PDA для L_1 , A_2 -НКА для L_2



Строки t для $L_1 \cap L_2$:

$$Q: Q_{A_1} \times Q_{A_2}$$

$$q_0: \langle q_0(A_1), q_0(A_2) \rangle$$

$$F: \{ \langle q_1, q_2 \rangle \mid q_1 \in F_{A_1} \text{ и } q_2 \in F_{A_2} \}$$

$$\Gamma: \Gamma(A_1)$$

Для переходов вида $\langle q_i, A \rangle \xrightarrow{\gamma} \langle q_j, \varphi \rangle (\gamma \in \Sigma) \in \delta(A_1)$
 $\langle q_k \rangle \xrightarrow{\gamma} q_m \in \delta(A_2)$
 $\Rightarrow \langle \langle q_i, q_k \rangle, A \rangle \xrightarrow{\gamma} \langle \langle q_j, q_m \rangle, \varphi \rangle$

Для переходов вида $\langle q_i, A \rangle \xrightarrow{\epsilon} \langle q_j, \varphi \rangle \in \delta(A_1)$ и вида $q_k \in Q(A_2)$
 добавим $\langle \langle q_i, q_k \rangle, A \rangle \xrightarrow{\epsilon} \langle \langle q_j, q_k \rangle, \varphi \rangle$

Для того, чтобы сохр. структуру правил CFG при ли-ии с НКА, изменения добавим нетерм.:

$$\{ \langle q_i, A_i, q_k \rangle \mid q_i, q_k \in Q(\text{НКА}) \text{ и } A_i \in N \}$$

$$t_{ij} \rightarrow \underbrace{B_1 \dots B_k}_{q_1 \xrightarrow{\gamma} q_2 \xrightarrow{\gamma} q_3 \dots \xrightarrow{\gamma} q_{k+1}}$$

Для переходы $q_i \xrightarrow{\gamma} q_j (\gamma \in \Sigma)$ и правилу $t_k \rightarrow \gamma$ добавим
 $\langle q_i, A_k, q_j \rangle \rightarrow \gamma$

Для правилу $t_k \rightarrow B_1 \dots B_N$ ($B_i \in N$) и вида возможных $q, q_1, q_2 \dots q_N \in Q(\text{НКА})$ добавим

$$\langle q_0, \underline{t_k}, q_N \rangle \rightarrow \langle q_0, \overline{B_1}, q_1 \rangle \langle q_1, \overline{B_2}, q_2 \rangle \dots \langle q_{N-1}, \overline{B_N}, q_N \rangle$$

Предположим, что $A_i \rightarrow A_j A_k$ $A_i \rightarrow \gamma$ ($\gamma \in \Sigma$)

символы a_1, \dots, a_n

start \rightarrow	1	2	\dots	k
1				
2				
3				
\vdots				
K				

} табл. при соотв-ии авт.

клетки $\langle i, j \rangle$ -иерархия метрик, расположенных отрезок слова $a_{i+1} \dots a_j$

1) $A_i \rightarrow a_n$ (правило \mathcal{F}), то заполнение клетки $\langle m, m \rangle$ метрик A_i :

2) $A \rightarrow BC$ и B стоит в позиции $\langle i, k \rangle$, C в позиции $\langle k+1, j \rangle$, то добав. в клетку $\langle i, j \rangle$ метрик A :

Алгоритм "СУК" — конструкция Г-ие помогает построить алгоритм распознавания произвольной CFG

[Если S старт $\beta \langle 1, k \rangle \iff$ разбор успешен.]

PDA: $\langle q_i, A \rangle \xrightarrow{\gamma} \langle q_j, \varphi \rangle$ $\Delta_{iA\gamma} (q_i, A, \gamma)$

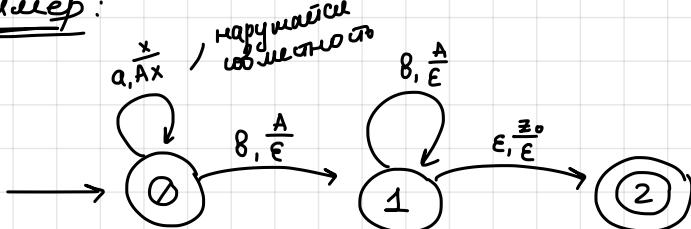
$$\begin{aligned} 1) & \left\{ \begin{array}{l} \langle q_i, A \rangle \xrightarrow{\overset{\gamma}{\cancel{E}}} \langle q_i, \Delta_{iA}\varphi \rangle \\ \langle q_i, \Delta_{iA} \rangle \xrightarrow{\gamma} \langle q_j, \varphi \rangle \end{array} \right. \text{аналог правила CFG} \\ 2) & \left\{ \begin{array}{l} \langle q_i, \Delta_{iA} \rangle \xrightarrow{\overset{\gamma}{\cancel{E}}} \langle q_j, \varphi \rangle \end{array} \right. \end{aligned}$$

Построим CFG G' с правилами: $A \rightarrow \Delta_{iA}\varphi$ $\Delta_{iA}\varphi \rightarrow \gamma$ + итер. соот-ии

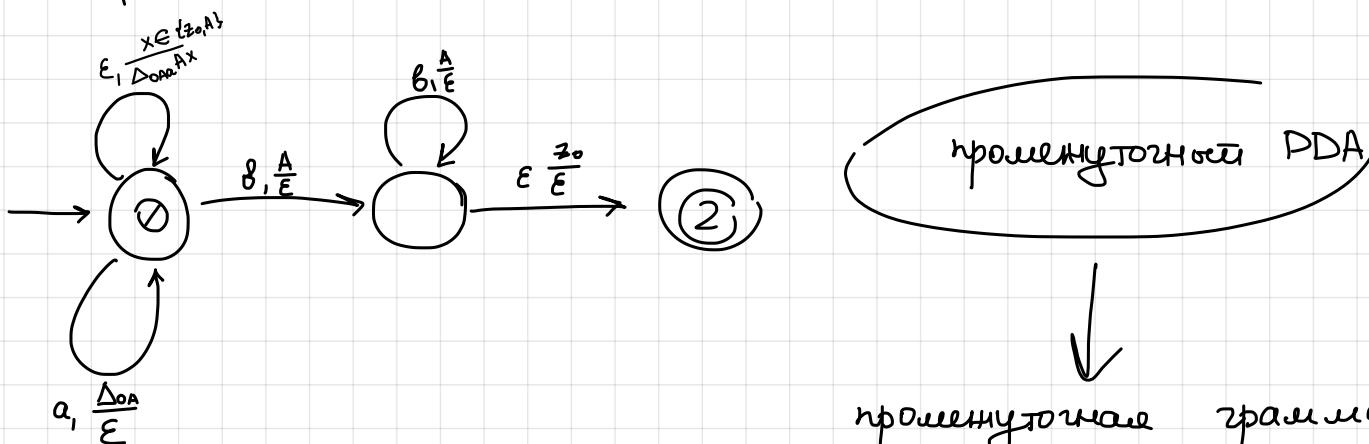
и DFA (НКА) с переходами $q_i \xrightarrow{\gamma} q_j$. Делаем пересечение, получим G

- Стартовое состояние грамматики G — \emptyset
- стартовое сост. DFA — q_0 (старт. сост. PDA)
- фин. сост. DFA — фин. сост. PDA

Пример:



$\{ a^n b^n \mid n > 0 \}$



промежуточные грамматика:

$$S \rightarrow \Delta_{0A} AS$$

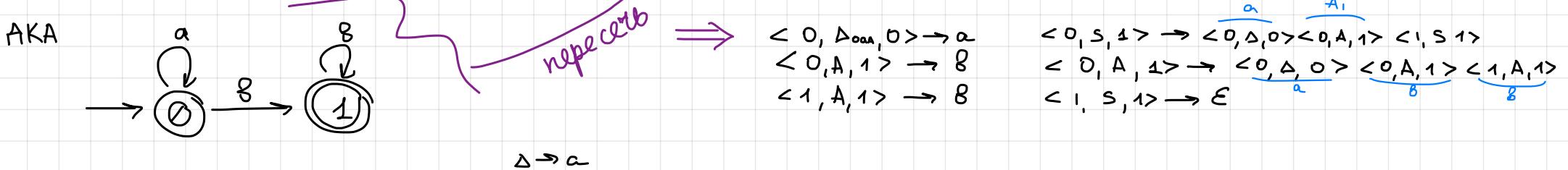
$$A \rightarrow \Delta_{0A} AA$$

$$A \rightarrow B$$

$$S \rightarrow E$$

} м.е. перенесли
все синт. действия
в грамматику

$$\Delta_{0A} \rightarrow a$$



III. o.
 $S \rightarrow a A_1$
 $A_1 \rightarrow B$
 $A_1 \rightarrow AA_1B$

$$\begin{aligned} <0, \Delta_{aa}, 0> &\rightarrow a \\ <0, A, 1> &\rightarrow B \\ <1, A, 1> &\rightarrow B \\ <1, S, 1> &\rightarrow \epsilon \end{aligned}$$

$$\begin{aligned} <0, S, 1> &\rightarrow <0, \Delta, 0> <0, A, 1> <1, S, 1> \\ <0, A, 1> &\rightarrow <0, \Delta, 0> <0, A, 1> <1, A, 1> \\ &\quad \text{with } \frac{a}{\Delta} \text{ and } \frac{A_1}{B} \end{aligned}$$

ЛЕКЦИЯ 7

ДЕРЕВЬЯ ВЫВОДА В КС-ГРАММАТИКЕ И ПРЕВЕСНЫЕ ЯЗЫКИ (I)

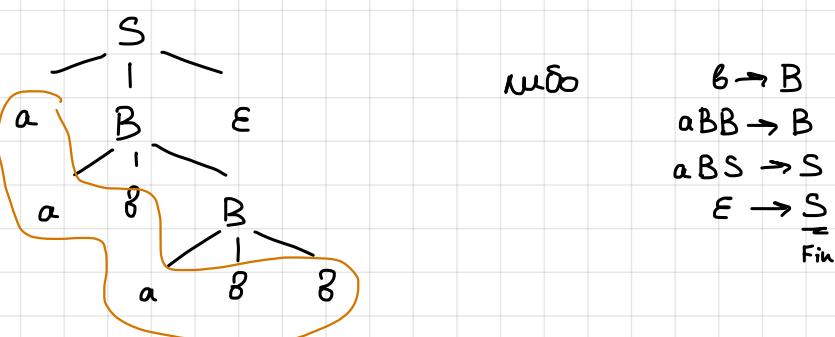
$\{N, \Sigma, S, R\}$ - КС грам., где $N \times (N \cup \Sigma)^*$
 / терм. / ин-во правил
 матер. / старт. мат. / ввода

$A \rightarrow \varphi_1 \dots \varphi_k$

$S \rightarrow aBS \mid \epsilon$

$B \rightarrow B \mid aBB$

$\not\vdash a a B a B B$



ибо

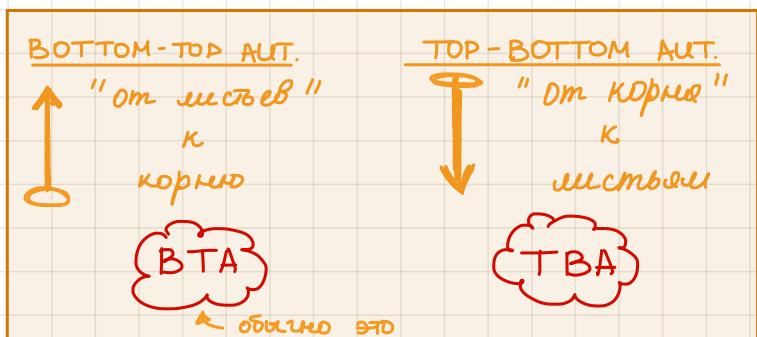
$B \rightarrow B$
 $aBB \rightarrow B$
 $aBS \rightarrow S$
 $E \rightarrow S$
 Fin

Дерево разбора

Definition: Дерево разбора - это дерево, в листьях ком. находятся терминалы, ветвление подчинено метарегулайции.

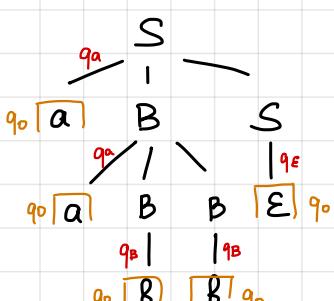
При этом ветвление подчиненное $A \in N$, может иметь только прямых потомков $\beta_1 \dots \beta_k$, где $A \rightarrow \beta_1 \dots \beta_k \in R$ и S -корень дерева.

Сигнатура - набор пар $\{<f_i, n_i> \mid f_i - \text{функциональность символов } \notin N \text{ и } n_i \in N - \text{арность}\}$



Конфигурация A: $T[S]$, в ком. есть $\square \leftarrow$ "дыра"

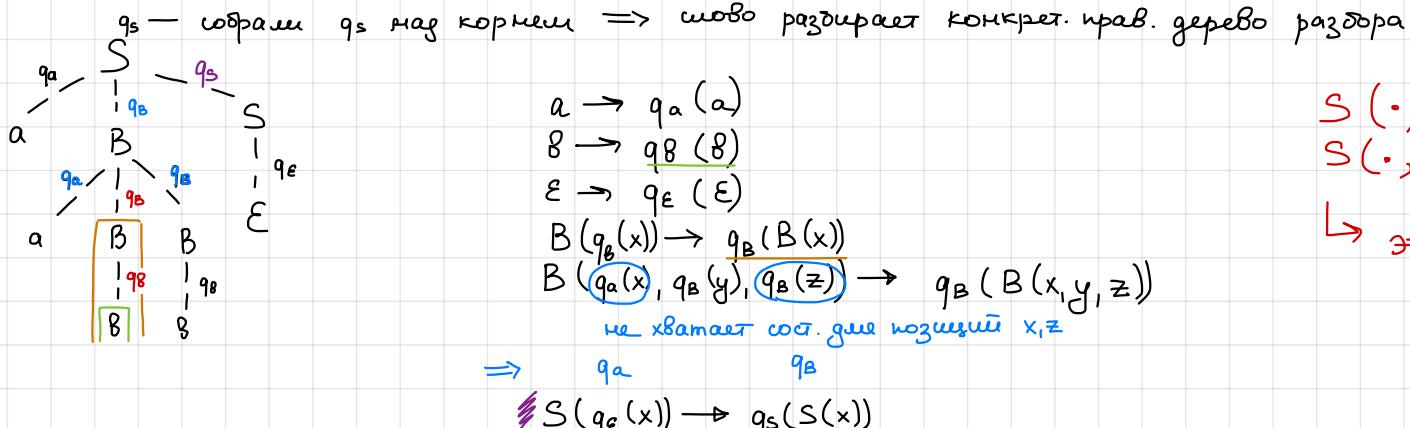
— т.е. еще то-то недописано



Со, м.р. $(C_0) = \emptyset$ — это стартовый конструктор BTA

$C_0 \rightarrow q_1(C_0)$ — синоним для перехода к листкам
 ↓ вписывание состояния ветвления

$f_k(q_1(x_1), \dots, q_k(x_k)) \rightarrow q_f(f_k(x_1, \dots, x_k))$



$a \rightarrow q_a(a)$
 $B \rightarrow q_B(B)$
 $E \rightarrow q_E(E)$
 $B(q_B(x)) \rightarrow q_B(B(x))$
 $B(q_a(x), q_B(y), q_B(z)) \rightarrow q_B(B(x, y, z))$
 $\Rightarrow q_a \quad q_B$
 $\# S(q_E(x)) \rightarrow q_S(S(x))$

$S(\cdot)$ — собирается одновременно
 $S(\cdot, \cdot, \cdot)$ — из 3-х эн.
 (с В тактие)
 ↳ это неоднозначность

Если в грамматике G есть правило вида $A_i \rightarrow \varphi_i \quad | \varphi_i| \neq | \varphi_j|$,
 $A_j \rightarrow \varphi_j$

тогда говорят членами. $|A_i| \varphi_i | A_j| \varphi_j$, используемое произвольно вместо A_i и
именование синтаксических $\langle A_i | \varphi_i \rangle, \langle A_j | \varphi_j \rangle$.

$$|\varphi_i| = 1 \quad |\varphi_j| = \frac{|\varphi_i|}{\varphi_i \neq E}$$

[если длина прав. $\varphi_i = 0$, то
 это нечто, назовем $|\varphi_i| = 1$]

$S \rightsquigarrow S_1, S_3 \Rightarrow S \rightarrow S_1 / S_3$
 $B \rightsquigarrow B_1, B_3 \Rightarrow S_1 \rightarrow E$
 $B_1 \rightarrow B$
 $B_3 \rightarrow aB_1, B_1 / aB_1, B_3 / aB_3, B_3$
 — все варианты

было $S \rightarrow aBS/E$
 $B \rightarrow B/aBB$

↳ это шаблон рука кости, кот. помогает избавиться от членов,
 кот. ищут разную аргент (т.е. возможность разных ветвлений)

Лемма

Грамматика G находится в нормальной форме Хомского \Leftrightarrow

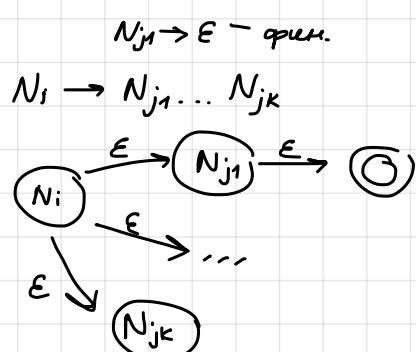
правильная часть G имеет вид: $A_i \rightarrow A_j A_k \quad (A_{ijk} \in N)$ или $A_i \rightarrow c \quad (c \in \Sigma)$ т.е. длина 2
или 1 и нет \times

Н.Ф. Хомского

Как устраним членное правило?

$A \rightarrow E$ (как избавиться?)

$A_1 \rightarrow A_2 A_3 \Rightarrow A_1 \rightarrow *E$
 $A_2 \rightarrow A_3 A_3 \Rightarrow A_2 \rightarrow *E$
 $A_3 \rightarrow E$



Посчитаем E -запись для A_i , если A_i порождает пустое слово, то

для всех правил вида $A_j \rightarrow \varphi, A_i \varphi_2$ добавляем $A_j \rightarrow \varphi, A_i \varphi_2 \cup A_j \rightarrow \varphi, \varphi_2$

не раскрыв-ся в E

Если $S \rightarrow *E$ имеет член E -зам., то введем самую стартовую S_0 и $S_0 \rightarrow E, S_0 \rightarrow S'$

и S убираем. Стераем все E правила, кроме $S_0 \rightarrow E$.

$\cancel{\exists} \quad S \rightarrow S_0 S B S / E :$

$S_0 \rightarrow E$

$S_0 \rightarrow S'$

$S' \rightarrow S'_a S'_b S' / aS'_b S' / S'_a b S' / S'_a S'_b / aS'_b / ab S' / ab$

ЛЕКЦИЯ 8

ДЕРЕВЬЯ РАЗБОРА И ДРЕВЕСНЫЕ ЯЗЫКИ (II)

$$A \rightarrow BC$$

$$A \rightarrow \varphi_1 a \varphi_2 \quad \text{G}_a$$

$$\varphi_1, \varphi_2 \neq \epsilon$$

$$G_a \rightarrow a$$

$A_i \rightarrow A_j$ — цепные правила (chain rules)

$A_j \in N$

$$A_i \xrightarrow{\epsilon} A_{i1} \dots \xrightarrow{\epsilon} A_{ik} \xrightarrow{\epsilon} A_j$$

$$A_i \rightarrow A_j \quad \Leftrightarrow \quad \frac{A_i \xrightarrow{\epsilon} A_j}{(A_i) \xrightarrow{\epsilon} (A_j)}$$

НКА

Как избавиться от цепных правил вида $A_i \rightarrow A_j$?

1) Строим замыкание отношения $\langle A_i, A_j \rangle \Leftrightarrow A_i \rightarrow A_j$

(НКА с ϵ -переходами)

Такое самое получаем все пары $\langle A_i, A_j \rangle \cong A_i \xrightarrow{*} A_j$, а далее постановка.

2) Во всех правилах $A_i \rightarrow \varphi_1 A_j \varphi_2$, где $|\varphi_1, \varphi_2| > 1$

Две все A_k : $A_j \xrightarrow{*} A_k$ строим $A_i \rightarrow \varphi_1 A_k \varphi_2$

3) Стираем все chain rules

Как избавиться от длинных правил?

Если правило R имеет вид $R: A_i \rightarrow B_1 \dots B_k$ ($k > 2$), то вводим соотв. нетерм. $I_1, I_2 \dots I_{k-2}$

и разбиваем правило: $A_i \rightarrow B_1 I_1$ и получаем Н.Р. Хомского

$$I_1 \rightarrow B_2 I_2$$

$$\dots$$

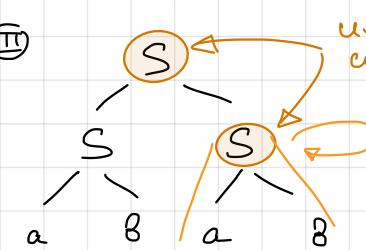
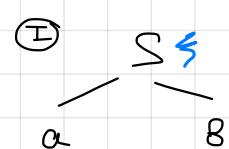
$$I_{k-2} \rightarrow B_{k-1} B_k$$

нет ϵ -правил	(✓)
нет chain rules	(✓)
нет сокращения Σ и N в н.р.	(✓)
нет длинн. правил	(✓)

} ХНР (н.р. Хомского)

✗ язык $\{S[S(a,b), S(a,b)], S[a,b]\}$ — язык древесный?

Состоит из 2-х деревьев



$$a \rightarrow q_a(a)$$

$$b \rightarrow q_b(b)$$

$$\$ S(q_a(x), q_b(y)) \rightarrow q_F(S(x,y))$$

$$S(q_F(x), q_F(y)) \xrightarrow{\text{X}} q_F(S(x,y))$$

$$S(q_F(x), q_F(y)) \rightarrow q_{F'}(S(x,y))$$

$$q_F, q_{F'} \in F$$

утил 3 S 8 Висоты

$$S(q_{F'}(x), q_a(y)) \rightarrow q_*(S(x,y))$$

— состоящее из узла

$$x \in t^*, F, F', a, b \}$$

Все изущенное содержание:

$$S(q_b(x), q_a(y)) \rightarrow q_*(S(x,y))$$

$$S(q_a(x), q_a(y)) \rightarrow q_* (S(x, y))$$

$$\alpha \in \{*, F, F^!, q, b\}$$

$$S(q_B(x), q_{F'}(y)) \rightarrow q_* (S(x, y))$$

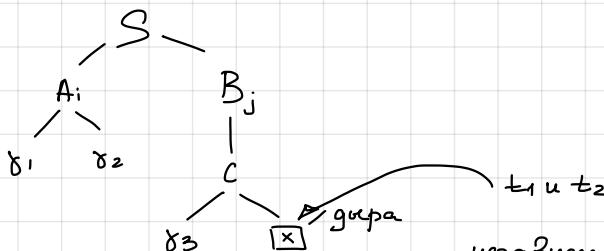
$$\beta \in \{ F', * \}$$

$$S(q_a(x), q_B(x)) \rightarrow q_* (S(x, y))$$

Теорема Матхисса - Нероде

Построение отношения на деревьях \equiv_r м.е. для любых деревьев с единственной переменной x (штабовой) $\exists u, v :$

$$t \equiv_r t_2 \Leftrightarrow \forall u[x] (u[x \rightarrow t_1] \in L \Leftrightarrow u[x \rightarrow t_2] \in L)$$

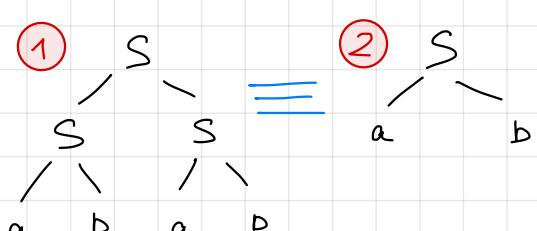


независимо от того, что мы подставим (t_1/t_2) не меняется расположение

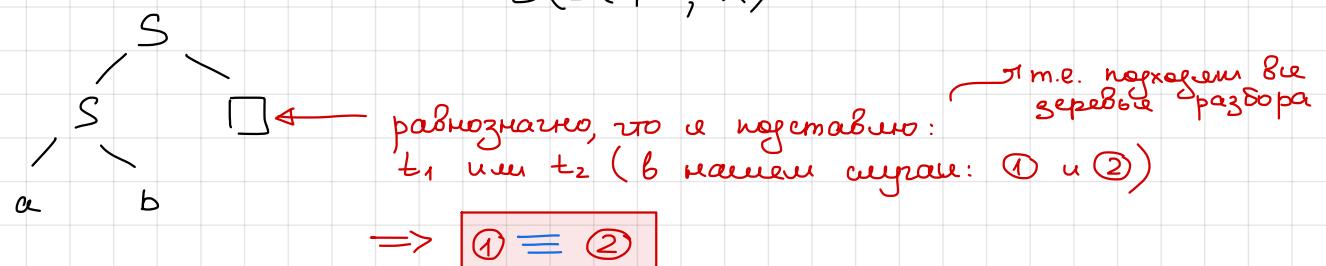
Теорема

L -древесно-автоматический $\Leftrightarrow \exists \equiv_r$ имеет конечн. число классов экв-ти

$\nabla: S \rightarrow SS \mid ab$



дерево выбора $S(S(a, b), X)$



Классы эквивалентности:

$\nabla: S \rightarrow SS \mid ab$

(1) Древесные разборы S

$\nabla: S \rightarrow SS \mid ab, S_0 \rightarrow S \mid T$ - старт.

(2) a

$T \rightarrow TT \mid ab \mid ST \mid SS \mid T$

(3) b

$S \rightarrow ST \mid TS \mid TT$

(4) TRAP

- $S \cup T$ попадают в один класс экв-ти

(1) Древесные разборы $S, T \rightsquigarrow$ биссектрирующие S, T

(0) Древесные разборы S_0

(2) a (3) b (4) TRAP

\Downarrow
 T, S взаимозависимы

$K \ni$ но \equiv_r — состоящие в ин. древесном автоматах

Следствие: как анализировать др. вида на древ. авт.



$$L = \{S(I^k(z_0), I^{k'}(z_0)) \mid k \in \mathbb{N}\} \quad - \text{автомат?}$$

смарт, не учитывая без ветвления
const

$\exists L$ - VPA, тогда $K \geq |\{\equiv_r\}|$

$$I(q_i(x)) \rightarrow$$

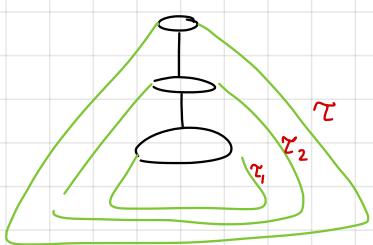
$$x \in K \ni i$$

$\not\equiv$ обозначает что встретятся 2 одинак. состояния при распознавании такого дерева ин. автомата

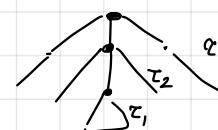
II.K. $T_2, T_1 \in L$ одному К.Э., то их можно заменить в нашем исход. дереве T_0 ,

m.e. $T_0(T_1 \rightarrow T_2) \in L$ \times - из-за прав. фатки

Также L - деревес-автоматический. Пусть $\tau \in L$ содержит путь длины $> |\{\equiv_\tau\}|$



Пусть τ_1, τ_2 из одного К.Э. $\tau_1 \subset \tau_2$
 $\tau_2 \rightarrow \tau_1 \in L$
 $(\tau_1 \rightarrow \tau_2) \in L$



Лемма о накачке для KC-алгебр

/оценка на число меток.

Если L - KC и $w \in L$, $|w| > N - \text{const}$ (N - число К.Э. в из. дерев. разбора - шт.)

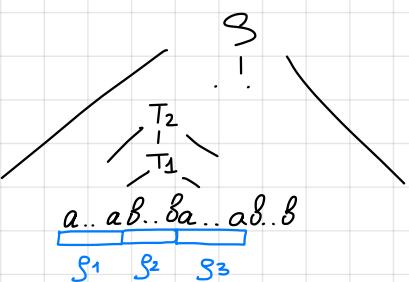
$$w = \underbrace{w_0 w_1}_{\text{start}} \underbrace{w_2 w_3}_{\tau_1} w_4 \quad \text{и} \quad \forall k \quad (w_0 w_1^k w_2 w_3^k w_4 \in L)$$

Если τ_2 и τ_1 симметричные близкие к штрафам, то $h(\tau_2) \leq K \cdot \text{высота} + 1$ и $|w_1 w_2 w_3| \leq N$ ($K = |\{\equiv_\tau\}|$)

✗ $\{ww \mid w \in \{a, b\}^*\}$ - KC? - нет

Множество длинеешее слово

$a^n b^n a^n b^n$, где $n > 2^{K+1}$ количество К.Э. М-Н. ВТА



① KC-алгебра замкнуты относ. reverse \Rightarrow
 T_2 выходит $S_1 - T_2$ выходит a^j , T_1 выходит a^i ($i < j$) $\Rightarrow S_1 < S_2$ и
 $S_2 < S_3$ при замене T_2 на T_1

T_2 содержит и "a", и "b", "b" в начале "a" в конце
 T_1 содержит либо что-то одно \Rightarrow не хватает
либо "a", "b" \rightarrow не хватает
 \Rightarrow расщеплено "a", "b" по 2-му блоку

но шагами выходит из цепочки \Rightarrow НЕ KC
и ∞ число К.Э.

PDA \cap DFA = PDA!

Чтобы сократить перебор парсов в w при док-де KC-авт, можно Л-ть L с пер. алгеброй

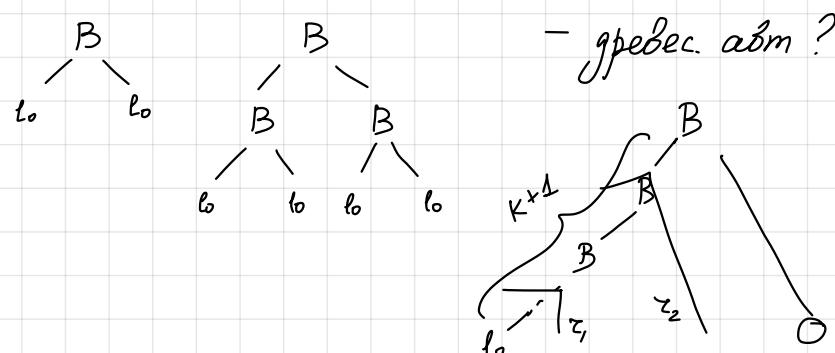
! Если L - KC, то $L \cap R$ - KC

✗ Если $\{ww \mid w \in \{a, b\}^*\} \cap a^+ b^+ a^+ b^+ \Rightarrow p_1/p_2$ и p_2/p_3 можно ли Л-ть м.к. выходит из пер.

Если τ , не на страже

✗ Таксо $L = \{B(t, t) \mid t \text{- бинарное дерево, все пути в ком. имеют одинаковую длину}\}$

B_2 - конструектор
 l_0 - штраф



$$\tau_1 \equiv_R \tau_2$$

! Взятие на 1 больше, чем К.Э. \Rightarrow только $\exists 2$ парсеров \in -ых 1-му К.Э

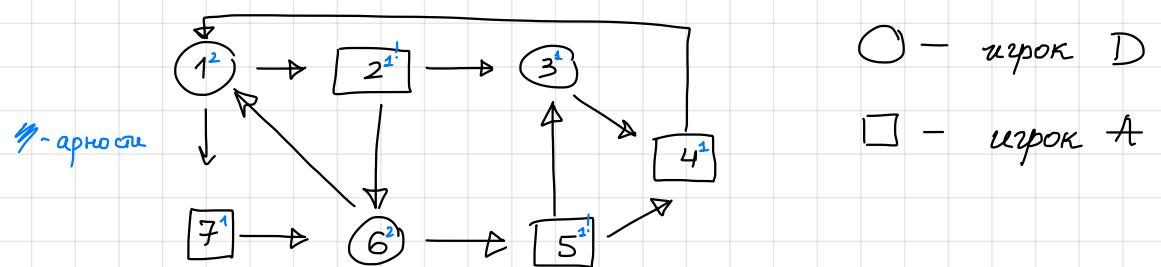
замена τ_2 на τ_1 выходит из. $\Rightarrow L$ - не деревес. авт.

ЛЕКЦИЯ 9

ПЕРЕКЛЮЧАЮЩИЕСЯ АВТОМАТЫ И ИГРЫ НА

КОНЕЧНЫХ ГРАФОВ

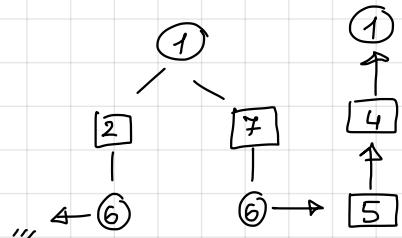
✗ конечный граф с трибунальной игрой:



○ - игрок D
□ - игрок A

Цель игрока D: не дать сост. 3 быть достигнутым
Цель игрока A: достичь сост. 3

③ - вопрос: где D



Получаемое, что игра описана деревом

Правила переходов для таких деревьев:

$$C(q_1(x_1) \dots q_n(x_n)) \xrightarrow{\text{конструктор}} q(C(x_1 \dots x_n)) \quad - \text{ ВТА}$$

TBA

$$q_{D_1}(1(x_1, x_2)) \xrightarrow{\text{конст.}} 1(q_{A_2}(x), q_{A_7}(x))$$

$$q_{A_2}(2(x)) \xrightarrow{} 2(q_{D_6}(x))$$

$$q_{D_3}(3(x)) \xrightarrow{} 3(\text{win})$$

$$q_{A_5}(5(x)) \xrightarrow{} 5(q_{A_4}(x))$$

$$q_{D_1}(1(x)) \xrightarrow{} 1(\text{lose})$$

$$q_{A_4}(4(x)) \xrightarrow{} 4(q_{D_1}(x))$$

$$q_{D_8}(6(x, y)) \xrightarrow{} 6(\text{lose}, \text{lose})$$

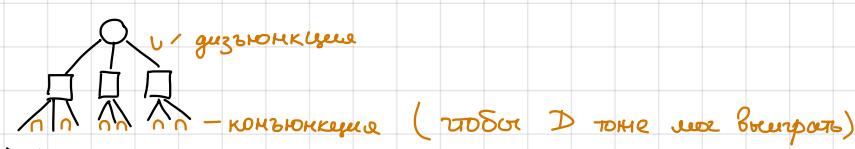
$$q_{A_7}(\gamma(x)) \xrightarrow{} \gamma(q_{D_6}(x))$$

$$q_{D_6}(6(x, y)) \xrightarrow{} 6(q_{D_1}(x), q_{A_5}(y))$$

Пусть $Q = Q_A \cup Q_D$, А может делать переход из Q_A , D - из Q_D

Пусть $F \subseteq Q$ - выигрыш. сост. для D

Правила перехода $\tilde{\delta} \subseteq Q \times Q$



AFA — Переключающийся автомат или Automating Finite Automaton

$$\left\{ \sum, Q, Q_V, F, q_0, \tilde{\delta} \right\} \quad (\Sigma \times \{\epsilon\}) \times Q \rightarrow Q$$

вс. сост. / F ⊆ Q

вход. алф. / мин-во сост.

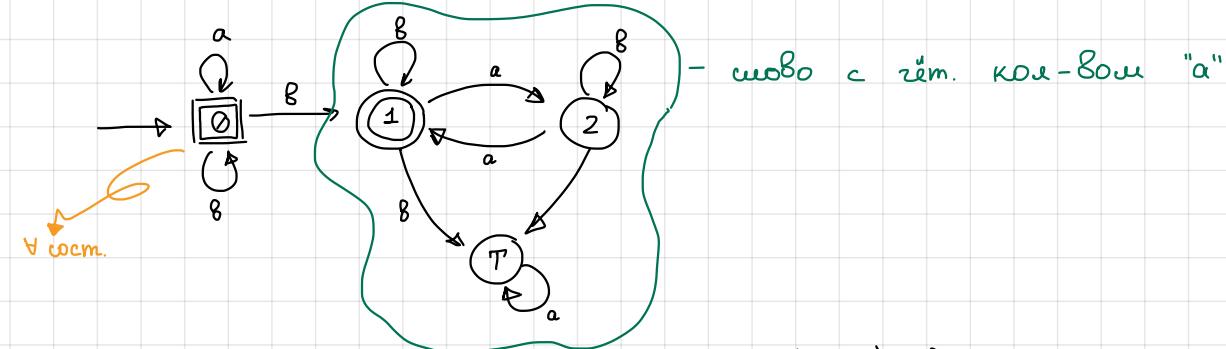
которые комьюнктивные (универсальные)

Слово и ж допускаемое AFA \iff

$$\exists \underbrace{q_0 \xrightarrow{\gamma} q_1 \xrightarrow{\gamma} \dots \xrightarrow{\gamma} q_n}_{w} : \text{если } q_i \in Q_V \text{ и } \frac{\text{путь}}{w}$$

компьюнкция имеет вид $\langle q_i, \underline{\delta_i} \rangle$ остается прописать супр.

то δ_i рассматривается по всем переходам из q_i



$(aab, \emptyset) \rightarrow (ab, \emptyset) \rightarrow (b, \emptyset) \xrightarrow{\begin{array}{l} (\varepsilon, 0) \\ (\varepsilon, a) \end{array}} \left\{ \begin{array}{l} \text{OK.} \\ \dots \end{array} \right.$

$(baab, \emptyset) \xrightarrow{\begin{array}{l} \text{контр.} \\ \text{гнезд.} \end{array}} (aab, 1) \xrightarrow{\begin{array}{l} \text{контр.} \\ \text{гнезд.} \end{array}} (ab, 2) \rightarrow (b, 1) \rightarrow (\varepsilon, 1) - \text{OK.}$

b^{aaaba}

НЕ OK т.к. 1 "a", а должно быть нет кол-во

$b^+ (aa)^* b^+ \frac{(aa)^*}{\text{зем}}$

Н.е. язык $a^* (b^+ (aa)^+)^* b^*$ — пер. язык.

↗ ba^n n -зем

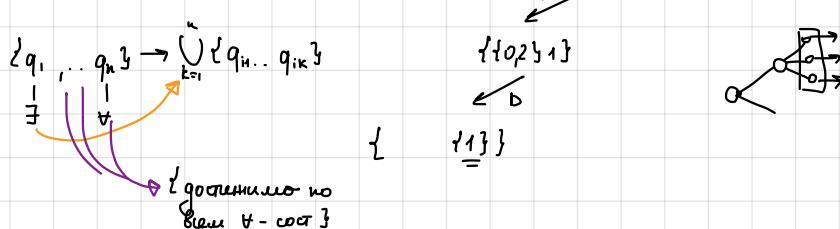
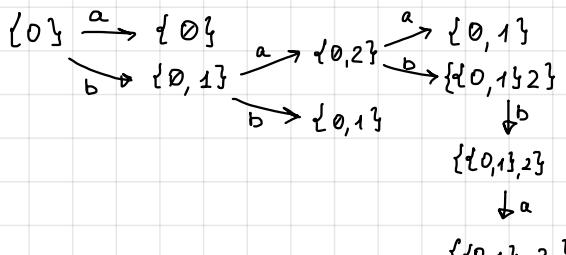
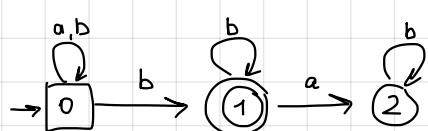


[] — квантор "на всех путях всё должно всп-ся"
 < > — квантор "хотя有一次 на одном"

Теорема про автоматаическое выражение

Языки, рассматриваемые АФА, решимые

+ кол-во (иначе ::)

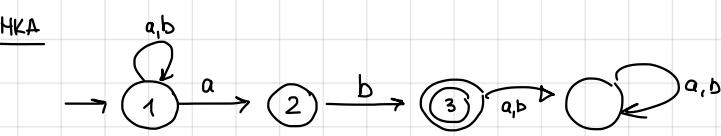


Утверждение

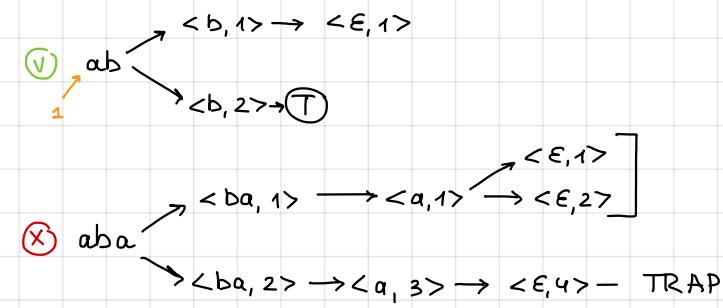
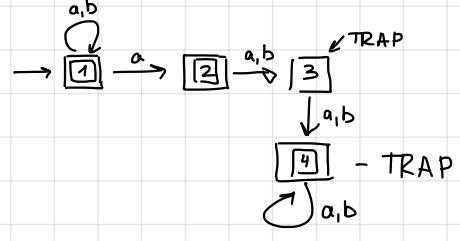
$A = \langle \Sigma, Q, Q_v, F, q_0, \delta \rangle$ автомат, расп. $\overline{L(A)}$ (дополнение языка A)

Обозн: \overline{A}

$\overline{A} = \langle \Sigma, Q, Q \setminus Q_v, Q \setminus F, q_0, \delta \rangle$

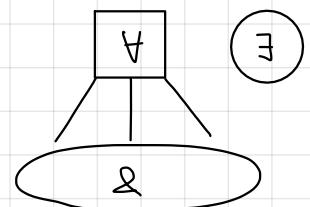


Donnerstag:



Л Е К Ц И Я 1 0

КОНЪЮНКТИВНЫЕ ГРАММАТИКИ



Компьютерное грамматики

Definition: $\langle \Sigma, N, S, \triangleright \rangle$ - комъюнкт. гр., если правила вывода

$$f_i \rightarrow \varphi_1 \wedge \dots \wedge \varphi_n \quad \varphi_j \in (\Sigma \cup N)^*$$

Если везде $\varphi_i = \gamma B_i / \gamma (\gamma \in \Sigma^*)$ и везде $\varphi_i = \gamma / B_i, \gamma$

AFA

Ме KC- өзгөкөн :

$$a^nb^nc^* \xrightarrow{S} B_1C \& \overline{AB_2} \xrightarrow{*} a^*b^nc^*$$

$$B \rightarrow \alpha B_1 \beta / \epsilon$$

$$C \rightarrow cC/\varepsilon$$

$$H \rightarrow \alpha\pi^+/\pi^-$$

$$2) \{ (a^n b)^m \mid n, m \in \mathbb{N} \}$$

$a^n b a^m b a^n \dots b a^n b$

$$\begin{aligned} S &\rightarrow G_1 \& G_2 \\ G_1 &\rightarrow PB\,G_1 \mid PB \\ P &\rightarrow aPa \mid b \\ G_2 &\rightarrow AB\,G_2' \mid AB \\ G_2' &\rightarrow PB\,G_2' \mid B \end{aligned}$$

$$\textcircled{1} \quad a^n b a^k b \quad ba^k b$$

\nwarrow \nearrow

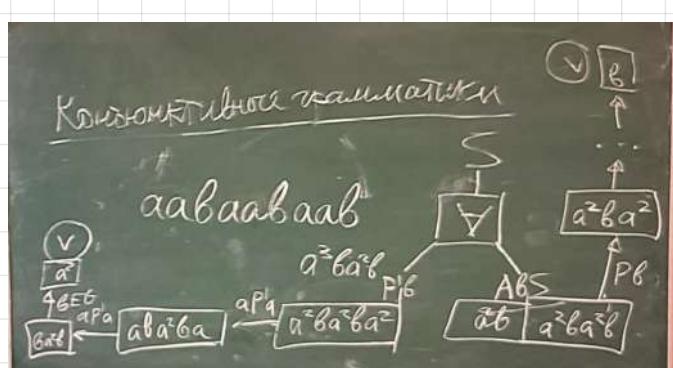
$$\begin{array}{c} S \rightarrow P8 \quad | \quad P'8 \quad \underline{\alpha} \quad A8S \\ P' \rightarrow \alpha P \alpha \quad | \quad 8^{(x)} \quad | \quad \beta E 8 \\ f \quad \alpha \quad A \quad | \quad S \end{array}$$

$$E \rightarrow aE \mid gE \mid e$$

$$P \rightarrow aP_0 \mid g$$

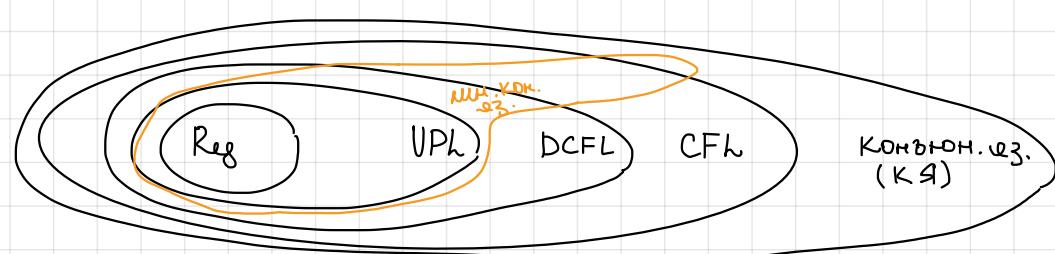
$$E \rightarrow \alpha E \mid \beta E$$

$$E \rightarrow \alpha E \mid \beta E$$



Ма КЯ переворачивается:

- 1) СΥК (Cocke - Yanger - Kasami)
 - 2) LL(K) - КЯ (ондег-се как, если все $A_i \rightarrow \varphi_j - LL(K)$, то $A_i \rightarrow \varphi_1 \& \dots \& \varphi_m - LL(K)$)

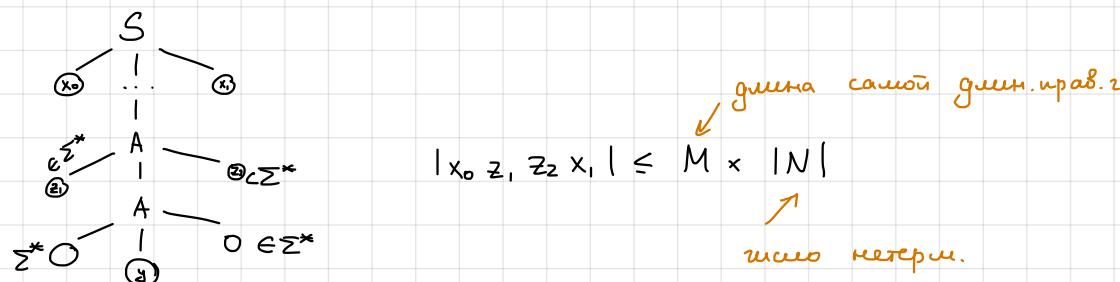


Линейные узлы — узлы, расч. CFL, в ком. Все правила сорти. не больше 1 метрик. в правой части

дерев. разб.:

множество терминальных

Возвращение близнейшего к корню подр. четвертичного



Лемма о макаке для линейных узлов

$$\text{L - univ. vlg.} \Rightarrow \exists p \left(\forall w \in L \left(|w| > p \Rightarrow \exists x_0, z_1, z_2, y, x_1 \left(w = x_0 z_1 y z_2 x_1 \text{ \& } |x_0 z_1 z_2 x_1| \leq p \text{ \& } |z_1 z_2| \neq e \right) \right) \right. \\ \left. \text{ \& } \forall k \in \mathbb{N} \left(x_0 z_1^k y z_2^k x_1 \in L \right) \right).$$

Абмонахов Григорий

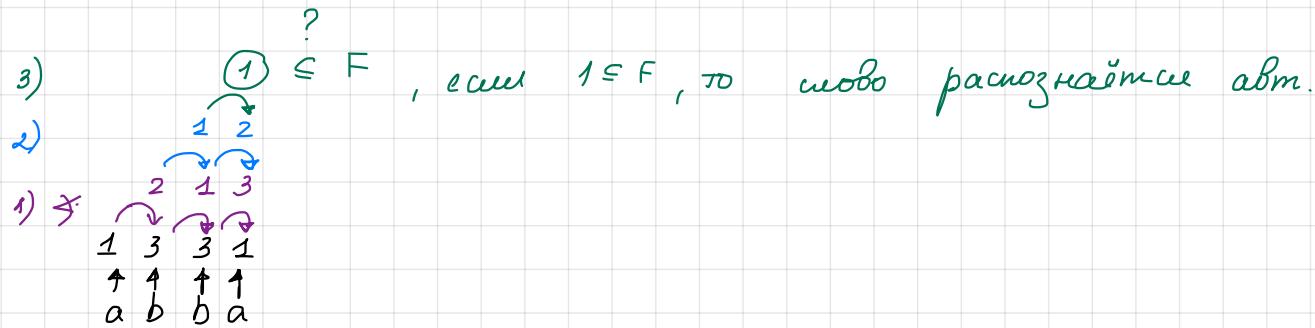
$$\begin{array}{c} < \sum, Q, F, S, I > \\ H \leq \sum_{\text{тепл.}} \times Q \end{array}$$

F ≤ Q

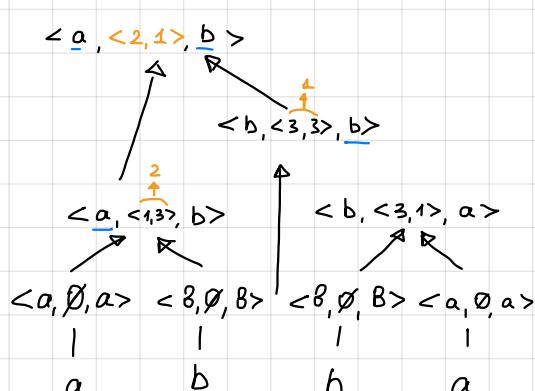
использованное правило

$\overline{S} \subseteq Q \times Q \rightarrow Q$																
$\underbrace{\hspace{100pt}}$ cupabá																
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td></td> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>2</td> </tr> <tr> <td>2</td> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <td>3</td> <td>3</td> <td>3</td> <td>1</td> </tr> </table>		1	2	3	1	1	1	2	2	1	2	3	3	3	3	1
	1	2	3													
1	1	1	2													
2	1	2	3													
3	3	3	1													

$$\underline{I} : \begin{matrix} a \rightarrow 1 \\ b \rightarrow 3 \end{matrix}$$



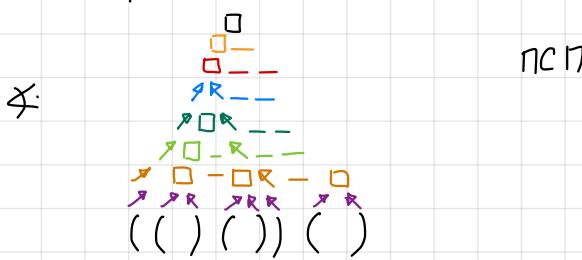
$1 \rightarrow a<2> \wedge <1>a$ — например $z-x$ вектор



АВТОМАТ ТРЕЛЛИСА = ЛИНЕЙНЫЕ КОНЪЮНКТ. ГРАММАТИКИ

	↗	↖	□	—
call	↗	↗	□	↗
ret	↖	—	↖	—
int	□	□	↖	□
—	—	↖	—	—

Интерпретация : $(\rightarrow \nearrow)$
 $) \rightarrow \nwarrow$
 $F = \{ \square \}$



ПСН

$\neq \{ w \in \{a,b\}^* \mid w \in \{a,b\}^* \text{ --- конъюнкт. линейно}$

$$S \rightarrow C \& D$$

$$C \rightarrow X C X \mid c \quad (a|b)^n c (a|b)^n$$

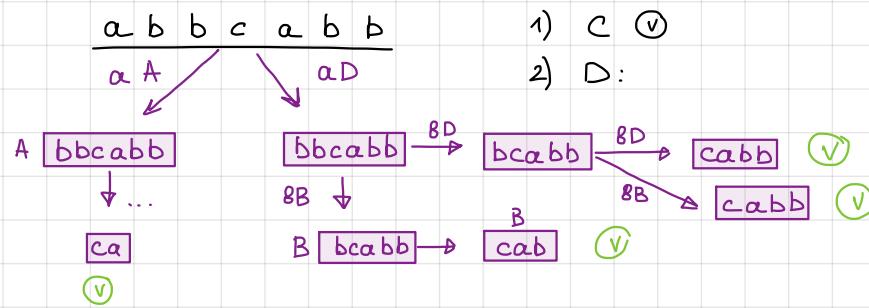
$$D \rightarrow a A \& a D \mid B B \& B D \mid c E$$

A $\rightarrow X A X \mid c E a$ — если б D видел "a", то заходит в рекур., кот. отсчитывает сколько букв сева и справа, пока не наткнется на

B $\rightarrow X B X \mid c E b$ буква "c": $a \underbrace{w_1}_c \underbrace{w_2}_b$ ($|w_1| = |w_2|$). Аналогично с "b"

X $\rightarrow a|B$ — просто # символ

E $\rightarrow X E \mid \varepsilon$ — everything str (дзг =)



DKA \rightarrow пер. яз. (comp. 122)