ЕЗИЦИ, АВТОМАТИ И ИЗЧИСЛИМОСТ

Записки за упражненията

Тодор Дуков

19 юни 2024 г.

СЪДЪРЖАНИЕ

ГЛАВА

1

ВЪВЕДЕНИЕ

Основното нещо, което се прави в този курс, е да се класифицират езици. За да можем да класифицираме един език, първо трябва знаем какво точно представлява един език.

1.1 Основни понятия

Дефиниция 1.1.1. Азбука ще наричаме всяко крайно множество. Елементите на азбуката ще наричаме **букви**.

Обикновено ще си бележим азбуквата със Σ. Също така, ако никъде не е споменато друго, $\Sigma = \{a, b\}$. Тук буквите ще бъдат a и b.

Дефиниция 1.1.2. Дума над азбуката Σ ще наричаме всяка крайна редица от букви от Σ. Дължината на дума α над Σ ще бележим с $|\alpha|$.

При азбуката $\Sigma = \{0, 1\}$, примерна дума ще бъде $\alpha = 000101$. Ясно е, че $|\alpha| = 6$.

Дефиниция 1.1.3. Празната дума ще наричаме единствената дума с дължина 0. Бележим я с ε.

Внимание. Важно е да се отбележи, че празното множество и празната дума са различни неща. Възможно е да се вземе такава дефиниция за редица, в която те да съвпадат, но това не е съществено. За нас думите ще бъдат едни неща, а множествата други. **TLDR:** $\varepsilon \neq \varnothing$

Дефиниция 1.1.4. Със Σ^* ще бележим множеството от всички думи над Σ. L ще наричаме **език** над Σ, ако $L \subseteq \Sigma$.

Тук нащият универсум ще бъде Σ^*. Така че за $L \subseteq \Sigma^*$, под $\overline{L}$ ще имаме предвид $\Sigma^* \setminus L$.

1.2 Операции върху думи и езици

Дефиниция 1.2.1. Ще дефинираме **конкатенацията** (слепването) на две думи α и β и ще го бележим с $\alpha \cdot \beta$

- $\alpha \cdot \varepsilon = \alpha$ (Базов случай)
- $\alpha \cdot (\beta x) = (\alpha \cdot \beta)x$ (Свеждане до по-малка "задача")

На пръв поглед такава дефиниция изглежда безсмислена, но това далеч не е така. После тя ще се използва постоянно в доказателства.

Нека разгледаме един пример за конкатенация:

$$aaa \cdot bbb = (aaa \cdot bb)b = ((aaa \cdot b)b)b = (((aaa \cdot \varepsilon)b)b)b = \\ = (((aaa)b)b)b = ((aaab)b)b = (aaabb)b = aaabbb$$

Вече можем да дефинираме α^n (n на брой пъти да конкатенираме думата α) индуктивно:

- $\alpha^0 = \varepsilon$
- $\alpha^{n+1} = \alpha^n \cdot \alpha$

За пример можем да вземем $(ab)^3$.

$$(ab)^3 = (ab)^2 \cdot ab = ((ab^1) \cdot ab) \cdot ab = (((ab^0) \cdot ab) \cdot ab) \cdot ab = \\ = ((\varepsilon \cdot ab) \cdot ab) \cdot ab = ab \cdot ab \cdot ab$$

Забележка. Тук използваме наготово, че $\varepsilon \cdot \alpha = \alpha$ (от Задача 1.4.2).

Имайки конкатенация на думи, можем да дефинираме и конкатенацията на езици. Най-естествено е да направим следното:

Дефиниция 1.2.2. Нека $L_1, L_2 \subseteq \Sigma^*$. Тогава **конкатенацията** на езиците L_1 и L_2 ще наричаме множеството:

$$L_1 \cdot L_2 = \{\alpha \cdot \beta \mid \alpha \in L_1 \ \& \ \beta \in L_2\}$$

Вече можем да дефинираме L^n (n на брой пъти да конкатенираме езика L) индуктивно:

- $L^0 = \{\varepsilon\}$
- $L^{n+1} = L^n \cdot L$

Дефиниция 1.2.3 (Звезда на Клини)**.** Нека $L \subseteq \Sigma^*$. Тогава:

- $L^* = \bigcup\limits_{n \in \mathbb{N}} L^n = L^0 \cup L^1 \cup L^2 \cup \ldots$
- $L^+ = \bigcup\limits_{\substack{n \in \mathbb{N} \\ n \neq 0}} L^n = L^1 \cup L^2 \cup L^3 \cup \ldots$

Ясно е, че в тази дефиниция Σ^* е същото нещо като в другата.

1.3 Допълнителни дефиниции

Тук ще сложим няколко дефиниции, които са стандартни, и ще има задачи, свързани с тях.

Дефиниция 1.3.1. Ще дефинираме **обръщането** на дума и на език. Обръщането на дума α, което бележим с α^{rev}, става индуктивно:

- $\varepsilon^{rev} = \varepsilon$
- $(\alpha x)^{rev} = x(\alpha^{rev})$

Обръщането на език L бележим с $L^{rev} = \{\alpha^{rev} \mid \alpha \in L\}$.

Дефиниция 1.3.2. Ще дефинираме кога една дума α е префикс, инфикс или суфикс на друга дума β:

- $\alpha \preceq_{pref} \beta$, ако $(\exists \gamma \in \Sigma^*)(\alpha \cdot \gamma = \beta)$
- $\alpha \preceq_{suff} \beta$, ако $(\exists \gamma \in \Sigma^*)(\gamma \cdot \alpha = \beta)$
- $\alpha \preceq_{inf} \beta$, ако $(\exists \gamma_1 \in \Sigma^*)(\exists \gamma_2 \in \Sigma^*)(\gamma_1 \cdot \alpha \cdot \gamma_2 = \beta)$

Нека $L \subseteq \Sigma^*$. Тогава:

- $\text{Pref}(L) = \{\alpha \in \Sigma^* \mid (\exists \beta \in L)(\alpha \preceq_{pref} \beta)\}$
- $\text{Suff}(L) = \{\alpha \in \Sigma^* \mid (\exists \beta \in L)(\alpha \preceq_{suff} \beta)\}$
- $\text{Infix}(L) = \{\alpha \in \Sigma^* \mid (\exists \beta \in L)(\alpha \preceq_{inf} \beta)\}$

1.4 Задачи за упражнение

Задача 1.4.1 (асоциативност)**.** *Да се докаже, че* $\alpha \cdot (\beta \cdot \gamma) = (\alpha \cdot \beta) \cdot \gamma$
Упътване: да се направи индукция по γ

Задача 1.4.2 (неутрален елемент)**.** *Да се докаже, че* $\varepsilon \cdot \alpha = \alpha$
Упътване: да се направи индукция по α

Задача 1.4.3. *Да се докаже, че* $\alpha^n \cdot \alpha^m = \alpha^{n+m}$
Упътване: да се направи индукция по m

Задача 1.4.4. *Да се докаже, че* $(\alpha^n)^m = \alpha^{nm}$
Упътване: да се направи индукция по m

Задача 1.4.5. *Да се докаже, че* $L^n \cdot L^m = L^{n+m}$
Упътване: да се направи индукция по m

Задача 1.4.6. *Да се докаже, че* $(L^n)^m = L^{nm}$
Упътване: да се направи индукция по m

Задача 1.4.7 (дистрибутивност)**.** *Да се докаже, че:*

- $(L_1 \cup L_2) \cdot L_3 = (L_1 \cdot L_3) \cup (L_2 \cdot L_3)$
- $(L_1 \cap L_2) \cdot L_3 = (L_1 \cdot L_3) \cap (L_2 \cdot L_3)$

Задача 1.4.8. *Да се докаже, че* $\Sigma^+ = \Sigma \cdot \Sigma^*$
Упътване: да се използват предните резултати

Задача 1.4.9 (свойства на reverse)**.** *Да се докаже, че:*

- $(\alpha \cdot \beta)^{rev} = \beta^{rev} \cdot \alpha^{rev}$
- $(L_1 \cdot L_2)^{rev} = L_2^{rev} \cdot L_1^{rev}$

Упътване: за първото да се направи индукция по β

Задача 1.4.10. *Да се докаже, че:*

- $(\alpha^{rev})^{rev} = \alpha$
- $(L^{rev})^{rev} = L$

Упътване: за първото да се направи индукция по α

Задача 1.4.11 (свойства на Pref, Suff, Infix)**.** *Да се докаже, че:*

- $\text{Pref}(L) = \text{Suff}(L^{rev})^{rev}$
- $\text{Suff}(L) = \text{Pref}(L^{rev})^{rev}$
- $\text{Pref}(L_1 \cdot L_2) = \text{Pref}(L_1) \cup (L_1 \cdot \text{Pref}(L_2))$
- $\text{Suff}(L_1 \cdot L_2) = \text{Suff}(L_2) \cup (\text{Suff}(L_1) \cdot L_2)$
- $\text{Infix}(L_1 \cdot L_2) = \text{Infix}(L_1) \cup \text{Infix}(L_2) \cup (\text{Suff}(L_1) \cdot \text{Pref}(L_2))$
- $\text{Infix}(L) = \text{Pref}(\text{Suff}(L)) = \text{Suff}(\text{Pref}(L))$

Упътване: да се разсъждава на ниво думи (конкатенация на езици се дефинира с конкатенация на думи)

ГЛАВА

2

АВТОМАТИ И РЕГУЛЯРНИ ИЗРАЗИ

Тук ще разгледаме първата "машина", с която ще класифицираме езиците.

2.1 Детерминирани автомати и автоматни езици

Дефиниция 2.1.1. Детерминиран краен автомат (накратко ДКА) ще наричаме всяко $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$, където:

- Σ е крайна азбука
- Q е крайно множество от състояния
- $s \in Q$ (ще го наричаме начално/стартово състояние)
- $\delta : Q \times \Sigma \to Q$ (ще я наричаме функция на преходите)
- $F \subseteq Q$ (ще ги наричаме финални състояния)

В момента със това, което имаме, ако искаме да покажем къде ще стигнем с думата aaa, започвайки от s, ще трябва да го запишем със $\delta(\delta(\delta(s,a),a),a)$, което е тромаво.
За това ще си въведем начин, по който да видим крайният резултат от прочитането на цяла дума.

Дефиниция 2.1.2. Дефинираме $\delta^* : Q \times \Sigma^* \to Q$ индуктивно:

- $\delta^*(p, \varepsilon) = p$ за всяко $p \in Q$
- $\delta^*(p, \beta x) = \delta(\delta^*(p, \beta), x)$ за всяко $p \in Q, \beta \in \Sigma^*, x \in \Sigma$

Сега можем да забележим, че:

$$\begin{aligned}\delta^*(s, aaa) = \delta(\delta^*(s, aa), a) = \delta(\delta(\delta^*(s, a), a), a) = \\ = \delta(\delta(\delta(\delta^*(s, \varepsilon), a), a), a) = \delta(\delta(\delta(s, a), a), a)\end{aligned}$$

Функцията наистина прави това, което искаме да прави.

Твърдение 2.1.3. $\delta^*(p, \alpha\beta) = \delta^*(\delta^*(p, \alpha), \beta)$

Доказателство. С индукция по $|\beta|$.

- $\delta^*(p, \alpha\varepsilon) = \delta^*(p, \alpha) = \delta^*(\delta^*(p, \alpha), \varepsilon)$
- $\delta^*(p, \alpha\beta x) = \delta(\delta^*(p, \alpha\beta), x) \overset{\text{ИП}}{=} \delta(\delta^*(\delta^*(p, \alpha), \beta), x) = \delta^*(\delta^*(p, \alpha), \beta x)$

□

Дефиниция 2.1.4. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е ДКА. Тогава езикът на автомата $\mathcal{A}$ ще наричаме множеството $\mathcal{L}(\mathcal{A}) = \{\alpha \in \Sigma^* \mid \delta^*(s, \alpha) \in F\}$. Един език $L \subseteq \Sigma^*$, наричаме **автоматен**, ако има ДКА $\mathcal{A}$ с $\mathcal{L}(\mathcal{A}) = L$.

2.2 Представяне на автомат

Ще видим начините, по които можем да представяме един автомат. За пример ще вземем $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$, където:

- $\Sigma = \{a, b\}$
- $Q = \{q_0, q_1, q_2\}$
- $s = q_0$
- $\delta(q_0, a) = q_2$
- $\delta(p, x) = q_1$ за $p \in Q$, $x \in \Sigma$, $\langle p, x \rangle \neq \langle q_0, a \rangle$
- $F = \{q_2\}$

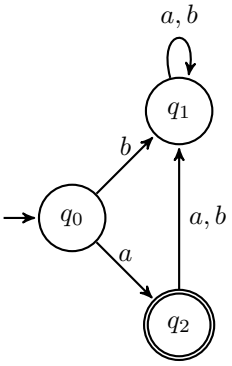
Това е първият начин за представяне. При него директно в явен вид се казват кои са всички съставни елементи на $\mathcal{A}$. Това ще бъде използване сравнително често, когато правим от един автомат друг. В случаите, в които не знаем как изглежда първоначалният автомат, следващите методи тогава няма да свършат работа.

Вторият начин за представяне е с таблица на функцията на преходите:

	a	b
$\rightarrow q_0$	q_2	q_1
q_1	q_1	q_1
$\checkmark q_2$	q_1	q_1

Тук с $\rightarrow$ отбелязваме началното състояние, а с $\checkmark$ отбелязваме финалните състояния. Във втората и третата колона казваме къде ще се озовем, ако се намираме в съответното състояние и прочетем съответната буква. Това е по-прегледно представяне от първото, но може да стане обемно.

Третият начин за представяне е с картинка:



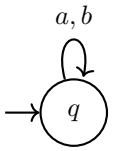
Началното състояние е отбелязано със стрелка, а финалните са оградени два пъти. Представянето чрез картинка изглежда по-прегледно от другите две, но то може и да стане огромно.
Ясно е, че $\mathcal{L}(\mathcal{A}) = \{a\}$. Ако искаме да покажем това формално, трябва да направим следното:

- Очевидно $a \in \mathcal{L}(\mathcal{A})$, $b \notin \mathcal{L}(\mathcal{A})$, $\varepsilon \notin \mathcal{L}(\mathcal{A})$
- Показваме, че $\delta^*(q_1, \alpha) = q_1$ за всяко $\alpha \in \Sigma^*$ с индукция по $|\alpha|$
- Ако $\alpha \notin \{a, b, \varepsilon\}$, то $\alpha = xy\beta$ за $x, y \in \Sigma$, $\beta \in \Sigma^*$
- Тогава лесно се вижда, че $\delta^*(q_0, \alpha) = q_1 \notin F$. Или директно отиваме в q_1 (ако $x = b$) и оставаме там, или първо отиваме в q_2 (ако $x = a$), и после със следващата буква отиваме в q_1 и оставаме там.

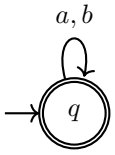
Това обаче за такива прости автомати не е нужно. Тези неща в такива случаи ще ги приемаме за очевидни.

2.3 Прости примери за автоматни езици

Нека видим много прости примери за автоматни езици, за да добием малко интуиция какви езици могат да бъдат автоматни, и да поработим малко със самите машини. Най-простите автоматни езици са $\varnothing$ и Σ^*:

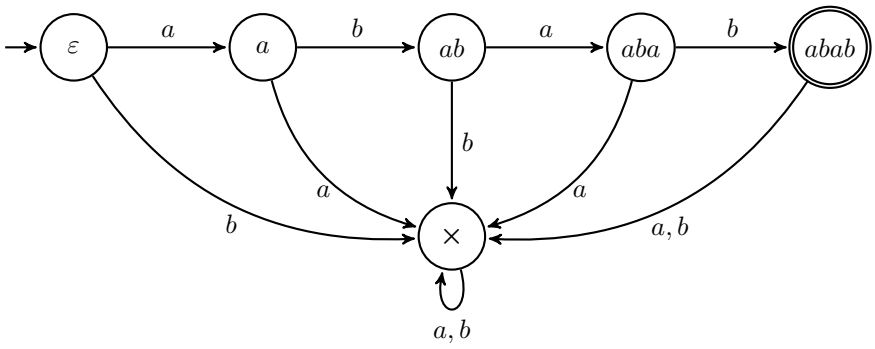


автомат за $\varnothing$



автомат за Σ^*

Сега малко ще усложним нещата. Ще направим автомат, който да разпознае езикът от една конкретна дума. Конструкцията за конкретната дума много лесно се обобщава за всички. За пример нека направим автомат за $L = \{abab\}$:

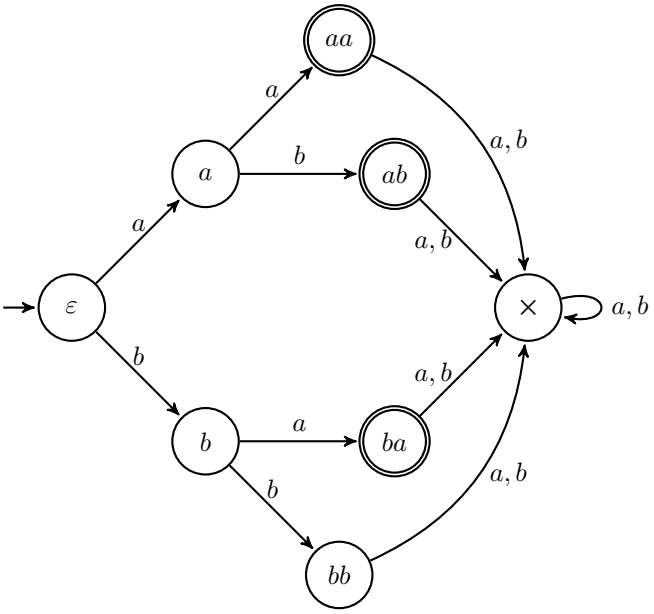


Тази техника може да се приложи за строене на автомат за език от произволно дълга дума. Ето как ще обобщим конструкцията за езика $L = \{\alpha\}$:

- $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$
- $Q = \text{Pref}(L) \cup \{\times\}$ (начален отрязък от пътя, който съставя α, $\times \notin \Sigma$ е отхвърлящо състояние боклук)
- $s = \varepsilon$
- За $\beta \preceq_{pref} \alpha$ (тогава $\beta \in Q$), $x \in \Sigma$: ако $\beta x \preceq_{pref} \alpha$, то $\delta(\beta, x) = \beta x$, иначе $\delta(\beta, x) = \times$
- $\delta(\times, x) = \times$ за $x \in \Sigma$
- $F = \{\alpha\}$

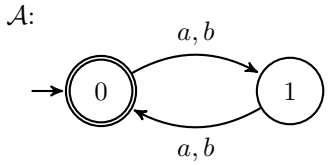
Идеята зад конструкцията е следната: Състоянията кодират пътя, който сме изминали, ако не сме се отклонили вече от "строежа" на α. В първият момент на отклонение отиваме във нефинално състояние "боклук", от което не може да излезнем.
Можем да разширим идеята, така че да работи за няколко думи като кодираме повече видове пътища. Ето как ще обобщим конструкцията (чиято коректност приемаме за очевидна) за $L = \{\alpha_1, \ldots, \alpha_n\}$:

- $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$
- $Q = \{\alpha \in \Sigma^* \mid (\exists \beta \in L)(|\alpha| \leq |\beta|)\} \cup \{\times\}$ (не гледаме пътища по-дълги от най-дългата дума в L, $\times \notin \Sigma$ - състояние боклук)
- $s = \varepsilon$
- За $\alpha \in Q$, $x \in \Sigma$: ако $\alpha x \in Q$, то $\delta(\alpha, x) = \alpha x$, иначе $\delta(\alpha, x) = \times$
- $\delta(\times, x) = \times$ за $x \in \Sigma$
- $F = L$



пример за прилагане на конструкцията за $L = \{ab, ba, aa\}$

Нека сега направим автомат за $L = \{\alpha \in \Sigma^* \mid |\alpha| \text{ е четно}\}$.
За една дума $\alpha \in \Sigma^*$ знаем, че тя или има четна дължина или има нечетна дължина. Дали не можем да кодираме по някакъв начин четността на прочетената дума в състояние? Отговорът е, че можем. Автоматът е следния:



Знаем, че $|\varepsilon|$ е четно. За всяко $\alpha \in \Sigma^*$, $x \in \Sigma$ знаем, че $|\alpha x|$ има различна четност от $|\alpha|$. Така ние започваме с думата ε и 0 като четност на думата и за всяка буква сменяме четността.
Нека сега помислим как да докажем, че $\mathcal{L}(\mathcal{A}) = L$. За това ще трябва да покажем, че започвайки от 0 и четейки α ние наистина получаваме четността на $|\alpha|$ като състояние.

Твърдение 2.3.1. *За всяко $\alpha \in \Sigma^*$: $\delta^*(0, \alpha) = |\alpha| \ (mod\, 2)$*

Доказателство. С индукция по $|\alpha|$.

- $|\alpha| = 0$, тогава $\alpha = \varepsilon$. Наистина $\delta^*(0, \varepsilon) = 0 \ (mod\, 2)$ ✓
- $|\alpha| = n + 1$, тогава $\alpha = \beta x$ за $\beta \in \Sigma^*, |\beta| = n,\ x \in \Sigma$. Тогава:

$$\delta^*(0, \beta x) = \delta(\delta^*(0, \beta), x) \overset{\text{ИП}}{=} \delta(|\beta| \ (mod\, 2), x) = |\beta| + 1 \ (mod\, 2) = |\beta x| \ (mod\, 2) = |\alpha| \ (mod\, 2)$$

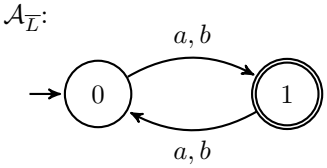
□

Състоянията наистина кодират информацията, която искахме. Имайки това можем да покажем, че двата езика съвпадат, по следния начин:

$$\alpha \in \mathcal{L}(\mathcal{A}) \iff \delta^*(0, \alpha) = 0 \iff |\alpha| \ (mod\, 2) = 0 \iff \alpha \in L$$

Нека сега помислим какъв автомат ще можем да направим за $\overline{L}$. Ние вече имаме автомат за L. От неговите състояние и преходи можем да извлечем информация за четността на дължината на думата. Единственото, което трябва да направим, е да сменим отговора на автомата. Ако преди той е казвал ДА, сега да казва НЕ, и обратното.

Това означава просто да обърнем финалните състояния:



Задача 2.3.2. *Да се построи автомат за:*

- $L_1 = \{a, b, ab\}$
- $L_2 = \{\varepsilon, ab, abab\}$
- $L_3 = \{aa, bb\}$

Упътване: за състояния да се префикси на думи от L_i

Задача 2.3.3. *Да се построи автомат за:*

- $L_1 = \Sigma^* \setminus \{a, b, ab\}$
- $L_2 = \Sigma^* \setminus \{\varepsilon, ab, abab\}$
- $L_3 = \Sigma^* \setminus \{aa, bb\}$

Упътване: да се използват автоматите от предната задача

Задача 2.3.4. *Да се построи автомат за:*

- $L_1 = \{\alpha \in \Sigma^* \mid |\alpha|$ *се дели на* $3\}$
- $L_2 = \{\alpha \in \Sigma^* \mid |\alpha|$ *се дели на* $5\}$
- $L_3 = \{\alpha \in \Sigma^* \mid |\alpha|$ *дава остатък* 2 *при деление на* $4\}$
- $L_4 = \{\alpha \in \Sigma^* \mid |\alpha|$ *дава остатък* 3 *при деление на* $5\}$

Упътване: за състояния да се използват остатъци при деление на n

Задача 2.3.5. *Да се построи автомат за:*

- $L_1 = \{\alpha \in \Sigma^* \mid |\alpha|$ *не се дели на* $3\}$
- $L_2 = \{\alpha \in \Sigma^* \mid |\alpha|$ *не се дели на* $5\}$
- $L_3 = \{\alpha \in \Sigma^* \mid |\alpha|$ *не дава остатък* 2 *при деление на* $4\}$
- $L_4 = \{\alpha \in \Sigma^* \mid |\alpha|$ *не дава остатък* 3 *при деление на* $5\}$

Упътване: да се използват автоматите от предната задача

2.4 Операции над автоматни езици

Сега ще разгледаме няколко операции, които запазват автоматност. Първата операция (която вече загатнахме) над автоматни езици, която ще разгледаме е допълнение.

Твърдение 2.4.1. *Ако L е автоматен, то тогава и $\overline{L}$ също е автоматен.*

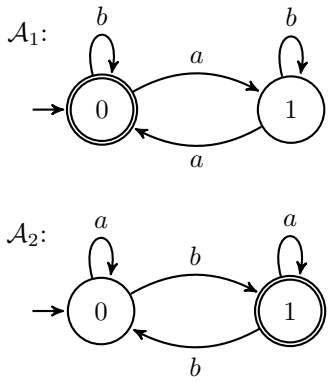
Доказателство. Понеже L е автоматен, има ДКА $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ с $\mathcal{L}(\mathcal{A}) = L$. Нека $\mathcal{A}_{\overline{L}} = \langle \Sigma, Q, s, \delta, Q \setminus F \rangle$. Тогава:

$$\alpha \in \mathcal{L}(\mathcal{A}) \iff \delta^*(s, \alpha) \in Q \setminus F \iff \delta^*(s, \alpha) \notin F \iff \alpha \notin L$$

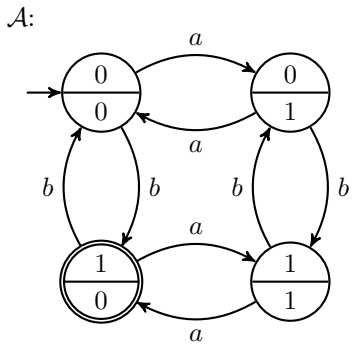
□

Както направихме и в предният пример, тук просто сменяме отговора. Състоянията и преходите на началният автомат ни дават достатъчна информация за структурата на думата, което е достатъчно за да кажем дали е от езика $\overline{L}$ или не е.

Преди да разгледаме следващата операция ще разгледаме два авомата:



Ще приемем за очевидно, че първият автомат разпознава думите с четен брой a, а вторият автомат разпознава думите с нечетен брой b.
Как бихме могли да направим автомат $\mathcal{A}$ за $\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$? Можем да направим така:



Имаме един брояч от за четността на a и още един брояч за четността на b. Всеки брояч се променя само от неговата буква. Накрая искаме отгоре да седи 1 (нечетен брой b), а отдолу да седи 0 (четен брой a).
Естествен въпрос, който човек може да си зададе, е дали това не може да се обобщи за произволни автомати. Оказва се, че може:

Твърдение 2.4.2. *Ако L_1 и L_2 са автоматни езици, то $L_1 \cap L_2$ също е автоматен език.*

Доказателство. Нека $\mathcal{A}_1 = \langle \Sigma, Q_1, s_1, \delta_1, F_1 \rangle$ е автомат за L_1 и нека $\mathcal{A}_2 = \langle \Sigma, Q_2, s_2, \delta_2, F_2 \rangle$ е автомат за L_2. Строим автомат за сечението:

- $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$
- $Q = Q_1 \times Q_2$
- $s = \langle s_1, s_2 \rangle$
- $\delta(\langle p_1, p_2 \rangle, x) = \langle \delta_1(p_1, x), \delta_2(p_2, x) \rangle$ за $\langle p_1, p_2 \rangle \in Q$, $x \in \Sigma$
- $F = F_1 \times F_2$

Тук използваме, че двата автомата, които имаме по начало ни дават информация за структурата на думите. Ние паралелно изпълняваме четене в $\mathcal{A}_1$ и $\mathcal{A}_2$, и накрая искаме положителен отговор и от двата автомата. Обаче ние трябва да покажем, че наистина двата автомата работят паралелно. Отново ни трябва някакво твърдение за функцията δ^*, която описва работата на автомата. Ние знаем, че за всяка буква правим преход и в двата компонента на текущото състояние $\langle p_1, p_2 \rangle$. Това, което искаме да проверим, е че същото нещо се случва при четенето на една дълга дума.

Ето го помощното твърдение:

Твърдение 2.4.3. $\delta^*(\langle p_1, p_2\rangle, \alpha) = \langle \delta_1^*(p_1, \alpha), \delta_2^*(p_2, \alpha)\rangle$

Доказателство. С индукция по $|\alpha|$.

- $\delta^*(\langle p_1, p_2\rangle, \varepsilon) = \langle p_1, p_2\rangle = \langle \delta_1^*(p_1, \varepsilon), \delta_2^*(p_2, \varepsilon)\rangle$ ✓
- $\delta^*(\langle p_1, p_2\rangle, \beta x) = \delta(\delta^*(\langle p_1, p_2\rangle, \beta), x) \overset{\text{ИП}}{=} \delta(\langle \delta_1^*(p_1, \beta), \delta_2^*(p_2, \beta)\rangle, x) = \langle \delta_1(\delta_1^*(p_1, \beta), x), \delta_2(\delta_2^*(p_2, \beta), x)\rangle = \langle \delta_1^*(p_1, \beta x), \delta_2^*(p_2, \beta x)\rangle$

□

Имайки това изкарваме директно, че:

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}) &\iff \delta^*(s, \alpha) \in F \\
&\iff \langle \delta_1^*(s_1, \alpha), \delta_2^*(s_2, \alpha)\rangle \in F_1 \times F_2 \\
&\iff \delta_1^*(s_1, \alpha) \in F_1 \,\&\, \delta_2^*(s_2, \alpha) \in F_2 \\
&\iff \alpha \in L_1 \,\&\, \alpha \in L_2 \\
&\iff \alpha \in L_1 \cap L_2
\end{aligned}$$

□

Имайки, че $\cap$ и допълнение запазват автоматност, директно изкарваме, че $\cup$ и $\setminus$ запазват автоматност:

- $L_1 \cup L_2 = \overline{\overline{L_1 \cup L_2}} = \overline{\overline{L_1} \cap \overline{L_2}}$
  Друг вариант това да се направи е да се приложи същата конструкция, със разликата че $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$
- $L_1 \setminus L_2 = L_1 \cap \overline{L_2}$
  Друг вариант това да се направи е да се приложи същата конструкция, със разликата че $F = F_1 \times (Q_2 \setminus F_2)$

Освен, че можем да изпълняваме няколко автомата паралелно, ние също така можем и да ги караме да се редуват.

Твърдение 2.4.4. *Ако L_1 и L_2 са автоматни езици, то и*

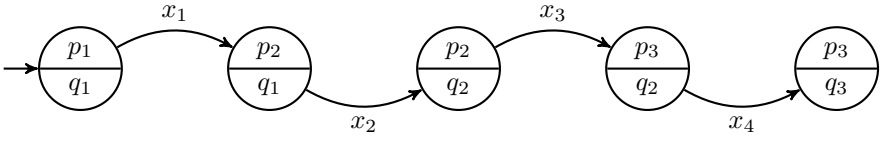
$$\mathrm{Mix}(L_1, L_2) = \{a_1 b_1 \dots a_n b_n \mid n \in \mathbb{N} \,\&\, (\forall i \in \{1, \dots, n\})(a_i \in \Sigma \,\&\, b_i \in \Sigma) \,\&\, a_1 \dots a_n \in L_1 \,\&\, b_1 \dots b_n \in L_2\}$$

е автоматен.

Доказателство. Нека $\mathcal{A}_1 = \langle \Sigma, Q_1, s_1, \delta_1, F_1\rangle$ е автомат за L_1 и нека $\mathcal{A}_2 = \langle \Sigma, Q_2, s_2, \delta_2, F_2\rangle$ е автомат за L_2. Строим автомат за $\mathrm{Mix}(L_1, L_2)$:

- $\mathcal{A} = \langle \Sigma, Q, s, \delta, F\rangle$
- $Q = Q_1 \times Q_2 \times \{0, 1\}$ (третата компонента ще ни казва с кой автомат работим в момента)
- $s = \langle s_1, s_2, 0\rangle$ (чисто начало, контрол има $\mathcal{A}_1$)
- $\delta(\langle p_1, p_2, 0\rangle, x) = \langle \delta_1(p_1, x), p_2, 1\rangle$ за $\langle p_1, p_2\rangle \in Q_1 \times Q_2$, $x \in \Sigma$
  Има контрол е $\mathcal{A}_1$, ние правим преход само с неговото състояние, и след това предаваме контрола на $\mathcal{A}_2$
- $\delta(\langle p_1, p_2, 1\rangle, x) = \langle p_1, \delta_2(p_2, x), 0\rangle$ за $\langle p_1, p_2\rangle \in Q_1 \times Q_2$, $x \in \Sigma$
  Аналогично тук има контрол е $\mathcal{A}_2$, затова ние правим преход само в него, и след това има контрол $\mathcal{A}_1$
- $F = F_1 \times F_2 \times \{0\}$ ($\mathcal{A}_1$ и $\mathcal{A}_2$ са одобрили своите думи и сме прочели дума с четна дължина)

Случва се нещо като в тази картинка:



Този автомат просто чете буква със $\mathcal{A}_1$ и буква със $\mathcal{A}_2$, после пак буква със $\mathcal{A}_1$ и буква със $\mathcal{A}_2$ и така нататък. Нека сега докажем, че това наистина се случва.

Твърдение 2.4.5. *За всяко $\alpha \in \Sigma^*$:*

- *ако $\alpha = \alpha_1 \dots \alpha_{2n}$ за някои $n \in \mathbb{N}$, $\alpha_1, \dots, \alpha_{2n} \in \Sigma$, то*
  $\delta^*(\langle s_1, s_2, 0\rangle, \alpha) = \langle \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n-1}), \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n}), 0\rangle$
- *ако $\alpha = \alpha_1 \dots \alpha_{2n+1}$ за някои $n \in \mathbb{N}$, $\alpha_1, \dots, \alpha_{2n+1} \in \Sigma$, то*
  $\delta^*(\langle s_1, s_2, 0\rangle, \alpha) = \langle \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n+1}), \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n}), 1\rangle$

Доказателство. С индукция по $|\alpha|$.
Базата е ясна. Нека разгледаме индукционната стъпка:

$$\begin{aligned}
&\delta^*(\langle s_1, s_2, 0\rangle, \alpha_1 \dots \alpha_{2n}) = \delta(\delta^*(\langle s_1, s_2, 0\rangle, \alpha_1 \dots \alpha_{2n-1}), \alpha_{2n}) \overset{\text{ИП}}{=} \\
&= \delta(\langle \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n-1}), \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n-2}), 1\rangle, \alpha_{2n}) = \\
&= \langle \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n-1}), \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n}), 0\rangle
\end{aligned}$$

$$\begin{aligned}
&\delta^*(\langle s_1, s_2, 0\rangle, \alpha_1 \dots \alpha_{2n+1}) = \delta(\delta^*(\langle s_1, s_2, 0\rangle, \alpha_1 \dots \alpha_{2n}), \alpha_{2n+1}) \overset{\text{ИП}}{=} \\
&= \delta(\langle \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n-1}), \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n}), 0\rangle, \alpha_{2n+1}) = \\
&= \langle \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n+1}), \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n}), 1\rangle
\end{aligned}$$

С това сме готови. □

Имайки това:

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}) &\iff \delta^*(s, \alpha) \in F \\
&\iff \alpha = \alpha_1 \dots \alpha_{2n}\ (\alpha_i \in \Sigma)\ \&\ \delta_1^*(s_1, \alpha_1\alpha_3 \dots \alpha_{2n-1}) \in L_1\ \&\ \delta_2^*(s_2, \alpha_2\alpha_4 \dots \alpha_{2n}) \in L_2 \\
&\iff \alpha \in \operatorname{Mix}(L_1, L_2)
\end{aligned}$$

□

Твърдение 2.4.6. *Нека L_1 и L_2 са автоматни езици и $\# \notin \Sigma$. Тогава $L_1 \cdot \{\#\} \cdot L_2$ също е автоматен език.*

Доказателство. Ще покажем само конструкцията, а доказателството ще оставим на читателя.
Нека $\mathcal{A}_i = \langle \Sigma, Q_i, s_i, \delta_i, F_i\rangle$ е автомат за L_i (за $i = 1, 2$). Б.О.О. нека $Q_1 \cap Q_2 = \varnothing$. Също така нека $\times \notin \Sigma$. Автоматът за $L_1 \cdot \{\#\} \cdot L_2$ ще бъде $\mathcal{A} = \langle \Sigma, Q_1 \cup Q_2 \cup \{\times\}, s_1, \delta, F_2\rangle$, където:

- $\delta(p, x) = \delta_1(p, x)$ за $x \in \Sigma$, $p \in Q_1$
- $\delta(p, \#) = \times$ за $p \notin F_1$
- $\delta(f, \#) = s_2$ за $f \in F_1$
- $\delta(p, x) = \delta_2(p, x)$ за $x \in \Sigma$, $p \in Q_2$
- $\delta(p, \#) = \times$ за $p \in Q_2$
- $\delta(\times, x) = \times$ за $x \in \Sigma \cup \{\#\}$

Казано на естествен език, четем със $\mathcal{A}_1$ докато не стигнем #, после четем със $\mathcal{A}_2$. Накрая ще искаме да сме прочели # и прочетеното от $\mathcal{A}_1$ и $\mathcal{A}_2$ да бъде одобрено (т.е. да сме получили отговор ДА). □

Твърдение 2.4.7. *За всеки автоматен L езиците* $\operatorname{Pref}(L)$, $\operatorname{Suff}(L)$ *и* $\operatorname{Infix}(L)$ *също са автоматни.*

Доказателство. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F\rangle$ е автомат за L.
Нека $F' = \{q \in Q \mid (\exists \gamma \in \Sigma^*)\ (\delta^*(q, \gamma) \in F)\}$ и $S' = \{\delta^*(s, \beta) \mid \beta \in \Sigma^*\}$. Да помислим кога точно една дума е префикс на дума от L:

$$\begin{aligned}
\beta \in \operatorname{Pref}(L) &\iff (\exists \gamma \in \Sigma^*)\ (\beta\gamma \in L) \\
&\iff (\exists \gamma \in \Sigma^*)\ (\delta^*(s, \beta\gamma) \in F) \\
&\iff (\exists \gamma \in \Sigma^*)\ (\delta^*(\delta^*(s, \beta), \gamma) \in F) \\
&\iff \delta^*(s, \beta) \in F' \\
&\iff \beta \in \mathcal{L}(\langle \Sigma, Q, s, \delta, F'\rangle)
\end{aligned}$$

Току що показахме автомат за $\text{Pref}(L)$, а именно $\langle \Sigma, Q, s, \delta, F' \rangle$. Така получихме, че $\text{Pref}(L)$ е автоматен език. Можем да направим подобни разсъждения за суфикс:

$$\begin{aligned}
\gamma \in \text{Suff}(L) &\iff (\exists \beta \in \Sigma^*)\,(\beta\gamma \in L) \\
&\iff (\exists \beta \in \Sigma^*)\,(\delta^*(s, \beta\gamma) \in F) \\
&\iff (\exists \beta \in \Sigma^*)\,(\delta^*(\delta^*(s, \beta), \gamma) \in F) \\
&\iff (\exists q \in S')\,(\delta^*(q, \gamma) \in F) \\
&\iff \gamma \in \bigcup_{q \in S'} \mathcal{L}(\langle \Sigma, Q, q, \delta, F \rangle)
\end{aligned}$$

Представихме $\text{Suff}(L)$ като обединение на автоматни езици, откъдето и той е автоматен. Освен това $\text{Infix}(L) = \text{Pref}(\text{Suff}(L))$ (от Задача 1.4.11), с което сме готови. □

2.5 Недетерминирани автомати

Дефиниция 2.5.1. Недетерминиран краен автомат (или НКА) ще наричаме всяко $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$, където:

- Σ е крайна азбука
- Q е крайно множество от състояния
- $S \subseteq Q$ (ще ги наричаме начални/стартови състояния)
- $\Delta : Q \times \Sigma \to \mathcal{P}(Q)$ (ще я наричаме функция на преходите)
- $F \subseteq Q$ (ще ги наричаме финални състояния)

Цялата "недетерминираност" идва от това, че от едно състояние с една буква можем да отидем на няколко места. Поне на пръв поглед тази нова машина изглежда доста по-мощна от старата. Вместо да си налагаме точно един преход, можем да направим два или три прехода, а можем да отидем в "нищото" (има се предвид $\varnothing$).

Дефиниция 2.5.2. Дефинираме $\Delta^* : \mathcal{P}(Q) \times \Sigma^* \to \mathcal{P}(Q)$ индуктивно:

- $\Delta^*(P, \varepsilon) = P$ за всяко $P \subseteq Q$
- $\Delta^*(P, \beta x) = \bigcup_{q \in \Delta^*(P, \beta)} \Delta(q, x)$ за всяко $P \subseteq Q, \beta \in \Sigma^*, x \in \Sigma$

Твърдение 2.5.3. $\Delta^*(P, \alpha\beta) = \Delta^*(\Delta^*(P, \alpha), \beta)$

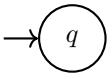
Доказателство. Елементарна индукция □

Вече можем да кажем какъв е език на даден недетерминиран автомат.

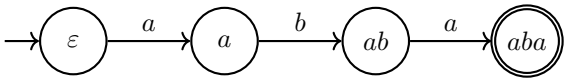
Дефиниция 2.5.4. Нека $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$ е НКА. Тогава езикът на автомата $\mathcal{N}$ е множеството:

$$\mathcal{L}(\mathcal{N}) = \{\alpha \in \Sigma^* \mid \delta^*(S, \alpha) \cap F \neq \varnothing\}.$$

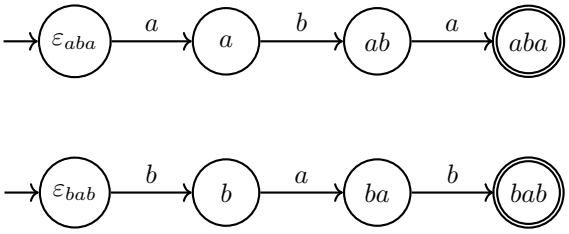
Тук цялата идея, е вместо думата да поеме по един предопределен път и да получи ДА или НЕ, тя да може да изпробва няколко възможни такива, и ако получи отговор ДА за поне един от тях, то тогава получава ДА от целия автомат. В някакъв смисъл има елемент на отгатване и несигурност. Може този път да свърши работа, но може и другия да свърши работа. Недетерминираните автомати са много добри, когато не сме сигурни какво точно искаме да се случи, но знаем всички възможности, с които да изпробваме. Тъй като ни трябва само един "клон" от изчислението да ни каже ДА, ние просто можем да изпробваме всички, които имат потенциал да ни свършат работа.
Нека дадем няколко прости примери за недетерминирани автомати:



Автомат за $\varnothing$



Автомат за $\{aba\}$



Автомат за $\{aba, bab\}$

Доста по-компактно става представянето на автомати за тези езици. Напълно елимираме ненужните преходи и състояние боклук. Можем и много лесно да направим автомат за няколко думи.
Тук се опитваме да познаем дали четем една от двете думи. Двата малки автомата работят независимо един от други и всеки дава собствен отговор. Ако сме се озовали във финално състояние, сме прочели една от двете думи в езика, иначе не сме.

2.6 Операции над езици на недетерминирани автомати

Тази конструкция може да се обобщи за обединение на всеки два езика на недетерминирани автомати.

Твърдение 2.6.1. *Нека $\mathcal{N}_i = \langle \Sigma, Q_i, S_i, \Delta_i, F_i \rangle$ е недетерминиран автомат за $i = 1, 2$. Тогава съществува недетерминиран автомат за езика $\mathcal{L}(\mathcal{N}_1) \cup \mathcal{L}(\mathcal{N}_2)$.*

Доказателство. Б.О.О. нека $\underbrace{Q_1 \cap Q_2 = \varnothing}_{(\star)}$. Тогава:

$$\begin{aligned}
\alpha \in \mathcal{L}(\langle \Sigma, Q_1 \cup Q_2, S_1 \cup S_2, \underbrace{\Delta_1 \cup \Delta_2}_{!!!}, F_1 \cup F_2 \rangle) &\iff (\Delta_1 \cup \Delta_2)(S_1 \cup S_2, \alpha) \cap (F_1 \cup F_2) \neq \varnothing \\
&\overset{(\star)}{\iff} \Delta_1^*(S_1, \alpha) \cap F_1 \neq \varnothing \vee \Delta_2^*(S_2, \alpha) \cap F_2 \neq \varnothing \\
&\iff \alpha \in \mathcal{L}(\mathcal{N}_1) \vee \alpha \in \mathcal{L}(\mathcal{N}_2) \\
&\iff \alpha \in \mathcal{L}(\mathcal{N}_1) \cup \mathcal{L}(\mathcal{N}_2)
\end{aligned}$$

Забележка (!!!). Понеже $(\star)$ е вярно, $\mathrm{Dom}(\Delta_1) \cap \mathrm{Dom}(\Delta_2) = \varnothing$, откъдето $\Delta_1 \cup \Delta_2$ е добре дефинирана функция.

□

Тази конструкция при недетерминирани автомати е по-компактна в сравнение с детерминираната версия. В този случай новите състояния са $|Q_1 \cup Q_2| = |Q_1| + |Q_2|$ на брой, докато при детерминираните автомати се получават $|Q_1 \times Q_2| = |Q_1| \cdot |Q_2|$ на брой състояния за новия автомат, което може да бъде много повече.

- $10 + 10 = 20$, докато $10 \cdot 10 = 100$
- $1000 + 2 = 1002$, докато $1000 \cdot 2 = 2000$
- $1000 + 1000 = 2000$, докато $1000 \cdot 1000 = 1000000$

Твърдение 2.6.2. *Нека L е автоматен език. Тогава има недетерминиран автомат за:*

$$\mathrm{ChangeSomeLetters}(L) = \{\alpha \in \Sigma^* \mid (\exists \beta \in L)\ (|\beta| = |\alpha|)\}$$

Доказателство. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е ДКА за L. Строим автомат $\mathcal{N}$ за $\mathrm{ChangeSomeLetters}(L)$:

- $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$
- $S = \{s\}$
- $\Delta(p, x) = \{\delta(p, a), \delta(p, b)\}$ за $p \in Q,\ x \in \Sigma$

Оставяме доказателството на следния факт на читателя, понеже е елементарна индукция:

$$\Delta^*(S, \alpha) = \{\delta^*(s, \beta) \mid \beta \in \Sigma\ \&\ |\beta| = |\alpha|\}$$

Имайки това твърдение получаваме, че:

$$\alpha \in \mathcal{L}(\mathcal{N}) \iff \Delta^*(S, \alpha) \cap F \neq \varnothing \iff (\exists \beta \in \Sigma^{|\alpha|})\,(\delta^*(s, \beta) \in F)$$
$$\iff (\exists \beta \in \Sigma^{|\alpha|})\,(\beta \in L) \iff \alpha \in \text{ChangeSomeLetters}(L)$$

□

Твърдение 2.6.3. *Нека $\mathcal{N}_i = \langle \Sigma, Q_i, S_i, \Delta_i, F_i \rangle$ е недетерминиран автомат за $i = 1, 2$. Тогава съществува недетерминиран автомат за езика $\mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2)$.*

Доказателство. Нека Б.О.О. $Q_1 \cap Q_2 = \varnothing$. Строим недетерминиран автомат $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$ за $\mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2)$:

- $Q = Q_1 \cup Q_2$
- $S = S_1 \cup S_2$ ако $\varepsilon \in \mathcal{L}(\mathcal{N}_1)$, иначе $S = S_1$
- $F = F_2$
- $\Delta(p, x) = \Delta_1(p, x) \cup S_2$, ако $\Delta_1(p, x) \cap F \neq \varnothing$ $(\star)$
- За другите състояния от Q_1 взимаме преходите от Δ_1, и аналогично за Q_2 взимаме същите преходи от Δ_2

Сега остава да покажем, че $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2)$.

- $\mathcal{L}(\mathcal{N}) \subseteq \mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2)$
  Нека $\alpha \in \mathcal{L}(\mathcal{N})$. Тогава $\Delta^*(S, \alpha) \cap F \neq \varnothing$. Ако сме стигнали до финално от S_2, то тогава $\alpha \in \mathcal{L}(\mathcal{N}_2)$ и $\varepsilon \in \mathcal{L}(\mathcal{N}_1)$, откъдето $\alpha = \varepsilon \cdot \alpha \in \mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2)$ и сме готови. В противен случай $\varepsilon \notin \mathcal{L}(\mathcal{N}_1)$ и има $\beta, \gamma \in \Sigma^*$ такива, че $\alpha = \beta\gamma$, $\beta \neq \varepsilon$, $S_2 \subseteq \Delta^*(S, \beta)$, т.е. сме приложили $(\star)$. Тогава с последната буква на β сме достигнали до финално състояние в $\mathcal{L}(\mathcal{N}_1)$, откъдето $\beta \in \mathcal{L}(\mathcal{N}_1)$. Също така понеже $Q_1 \cap Q_2 = \varnothing$, и състоянията от Q_2 имат само преходите на Δ_2, $\Delta(S_2, \gamma) \cap F \neq \varnothing$, откъдето $\gamma \in \mathcal{L}(\mathcal{N}_2)$. Така получаваме, че $\alpha = \beta \cdot \gamma \in \mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2)$.
- $\mathcal{L}(\mathcal{N}_1) \cdot \mathcal{L}(\mathcal{N}_2) \subseteq \mathcal{L}(\mathcal{N})$
  Нека $\beta \in \mathcal{L}(\mathcal{N}_1), \gamma \in \mathcal{L}(\mathcal{N}_2)$. Ако $\beta = \varepsilon$, то тогава $S_2 \subseteq S$, и понеже $\gamma \in \mathcal{L}(\mathcal{N}_2)$, $F = F_2$, то $\Delta^*(S, \underbrace{\beta \cdot \gamma}_{\gamma}) \cap F \neq \varnothing$, и от там $\beta \cdot \gamma \in \mathcal{L}(\mathcal{N})$. Ако $\beta \neq \varepsilon$, то $S_2 \subseteq \Delta^*(S, \beta)$, понеже $\beta \in \mathcal{L}(\mathcal{N}_1)$ и $(\star)$. Освен това и $\gamma \in \mathcal{L}(\mathcal{N}_2)$ т.е. $\Delta^*(S_2, \gamma) \cap F_2 \neq \varnothing$, откъдето $\Delta^*(S, \beta \cdot \gamma) \cap F \neq \varnothing$ т.е. $\beta \cdot \gamma \in \mathcal{L}(\mathcal{N})$.

□

Твърдение 2.6.4. *Нека $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$ е недетерминиран автомат. Тогава има недетерминиран автомат за езика $\mathcal{L}(\mathcal{N})^*$.*

Доказателство. Ще покажем само конструкцията, без да я обосноваваме (аналогична е на предната). Правим автомат за $\mathcal{L}(\mathcal{N})^+$ и използваме, че $L^* = L^+ \cup \{\varepsilon\}$ за всеки език L. Нека $\mathcal{N}' = \langle \Sigma, Q, S, \Delta', F \rangle$, където:

$$\Delta'(p, x) = \begin{cases} \Delta(p, x) & \text{ако } p \notin F \\ \Delta(p, x) \cup \Delta^*(S, x) & \text{ако } p \in F \end{cases} \tag{2.1}$$

Доказателството на коректност е аналогично на предната задача. □

Твърдение 2.6.5. *Нека $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$ е недетерминиран автомат. Тогава има недетерминиран автомат за езика $\mathcal{L}(\mathcal{N})^{rev}$.*

Доказателство. Нека $\mathcal{N}_{rev} = \langle \Sigma, Q, F, \Delta_{rev}, S \rangle$, където:

$$\Delta_{rev} = \{\langle p, x, q \rangle \mid p \in \Delta(q, x)\}$$

Тривиално може да се покаже с индукция, че за $\alpha \in \Sigma^*$, $p, q \in Q$:

$$q \in \Delta^*(\{p\}, \alpha) \iff p \in \Delta^*_{rev}(\{q\}, \alpha)\ (\star)$$

След това приключваме със:

$$\alpha \in \mathcal{L}(\mathcal{N}) \iff \Delta^*(S, \alpha) \cap F \neq \varnothing$$
$$\overset{(\star)}{\iff} \Delta^*_{rev}(F, \alpha) \cap S \neq \varnothing \iff \alpha \in \mathcal{L}(\mathcal{N}_{rev})$$

□

2.7 Еквивалентност на детерминирани и недетерминирани автомати

Въпреки че на пръв поглед недетерминираните автомати ни се струват по-мощни, те не ни дават нищо повече освен удобство. Първо ще покажем, по-очевидното твърдение.

Твърдение 2.7.1. *За всеки детерминиран автомат $\mathcal{A}$ съществува недетерминиран автомат $\mathcal{N}$ със $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A})$.*

Доказателство. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е детерминиран автомат. Нека $\mathcal{N} = \langle \Sigma, Q, \{s\}, \Delta, F \rangle$, където $\Delta(p, x) = \{\delta(p, x)\}$ за $p \in Q$. Тривиално може да се покаже с индукция, че за всяко $p \in Q$, $\alpha \in \Sigma^*$: $\Delta^*(p, \alpha) = \{\delta^*(p, \alpha)\}$. Така:

$$\begin{aligned} \alpha \in \mathcal{L}(\mathcal{N}) &\iff \Delta^*(\{s\}, \alpha) \cap F \neq \varnothing \iff \{\delta^*(s, \alpha)\} \cap F \neq \varnothing \\ &\iff \delta^*(s, \alpha) \in F \iff \alpha \in \mathcal{L}(\mathcal{A}) \end{aligned}$$

□

Сега пък ще покажем как "детерминизира" недетерминиран автомат.

Твърдение 2.7.2. *За всеки недетерминиран автомат $\mathcal{N}$ съществува детерминиран автомат $\mathcal{A}$ със $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{N})$.*

Доказателство. Нека $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$ е недетерминиран автомат. Нека $\mathcal{A} = \langle \Sigma, \mathcal{P}(Q), S, \delta, F' \rangle$, където $\delta(P, x) = \Delta^*(P, x)$ за $P \subseteq Q$ и $F' = \{P \in \mathcal{P}(Q) \mid P \cap F \neq \varnothing\}$. Тривиално може да се покаже с индукция, че за всяко $P \subseteq Q$, $\alpha \in \Sigma^*$: $\delta^*(P, \alpha) = \Delta^*(P, \alpha)$. Така:

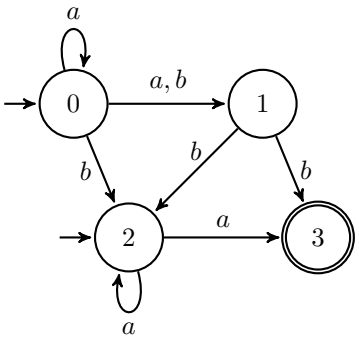
$$\begin{aligned} \alpha \in \mathcal{L}(\mathcal{N}) &\iff \Delta^*(S, \alpha) \cap F \neq \varnothing \iff \delta^*(S, \alpha) \cap F \neq \varnothing \\ &\iff \delta^*(S, \alpha) \in F' \iff \alpha \in \mathcal{L}(\mathcal{A}) \end{aligned}$$

□

Имайки това вече можем да използваме, че $\cdot$, $*$ и rev запазват автоматност.

Внимание. При "детерминизиране" броят на състоянията нараства експоненциално. Накратко 10 става 1024.

Задача 2.7.3. *Да се детерминира следният автомат:*

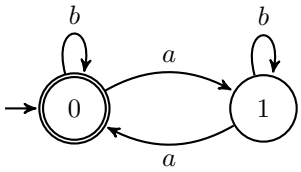


2.8 Друг начин да си мислим за автоматите

Освен като абстрактни машини, можем да си мислим за детерминираните автомати като за програми. Тези "програми" имат няколко много хубави свойства:

- винаги завършват работа
- работят с константна памет
- работят за линейно време спрямо дължината на дума

Нека вземем за пример следният автомат:



Ясно е, че той разпознава думи с четен брой букви a. Нека видим как бихме направили програма на C++, която приема низ и връща дали този низ съдържа четен брой a:

```
typedef char state_index;

const state_index table[2][2] = {{1, 0}, {0, 1}};   // таблица на преходите
const bool is_accepting_state[2] = { true, false }; // финални състояния
const state_index INITIAL_STATE = 0;                // начално състояние

// функция на преходите
state_index delta(state_index state, char letter) { return table[state][letter - 'a']; }

bool accepts_word(const std::string &word)
{
  // предполагаме валиден вход
  // т.е. низът съдържа само буквите 'a' и 'b'
  state_index state = INITIAL_STATE; // започваме в началното състояние

  for (size_t i = 0; i < word.size(); ++i)
  {
    state = delta(state, word[i]); // правим преход с настоящото състояние и буква
  }

  return is_accepting_state[state]; // накрая искаме да сме във финално състояние
}
```

Лесно можем да видим как тази конструкция може да се адаптира за произволен автомат $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, F \rangle$:

- Нека $Q = \{q_0, \dots q_{n-1}\}$. За състоянията се разбираме, че на q_i съответства числото i. За типа на `state_index` ще ни трябва променлива, която ще побере числата от 0 до $n-1$.
- Нека $\Sigma = \{\sigma_0, \dots, \sigma_{k-1}\}$. Нужна ни е функция, която да е биекция между Σ и $\{0, \dots, k-1\}$. Нека сигнатурата и да бъде `state_index letter_to_index(char letter)`. Също така ще искаме да работи в константно време, което е лесно, защото азбуката е с фиксиран размер.
- Правим `state_index table[n][k]` спрямо нашия оригинален автомат и кодировката от `letter_to_index`:

$$\texttt{table[i][letter\_to\_index(c)] == j е истина} \iff \delta^*(q_i, c) = q_j$$

- Правим `bool is_accepting_state[n]` спрямо нашия оригинален автомат и кодировката от `letter_to_index`:

$$\texttt{table[i] е истина} \iff q_i \in F$$

- Понеже q_0 е началното състояние:

  ```
  const state_index INITIAL_STATE = 0;
  ```

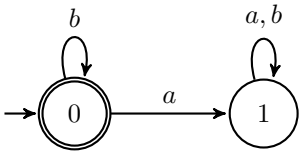
- Правим програмна реализация на δ чрез `table`:

  ```
  state_index delta(state_index state, char letter)
  {
      return table[state][letter_to_index(letter)];
  }
  ```

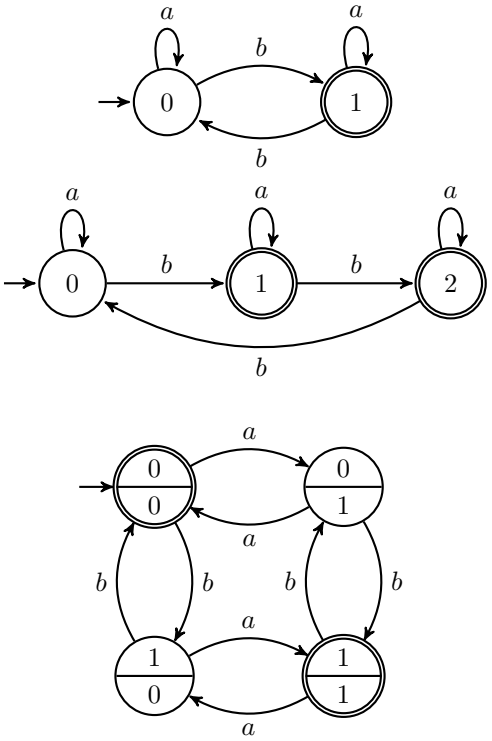
- Функцията `accepts_word` няма нужда от модификация. В зависимост от това дали има нужда може да се сложи валидация на входните данни.

Забележка. Разбира се това не е единственият възможен начин да се направи програмна реализация на автомат. За по-малко на брой състояния също би било удобно да се използват оператори `if` или `switch` със `case`. Обаче да имаме функцията δ записана по някакъв начин в таблица ни прави програмата по-бърза при повече състояния. Много по-лесно е да намериш елемент номер 10000 в масив отколкото да оцениш 10000 пъти оператор `if`.

Задача 2.8.1. *Да се направи реализация използвайки* `if` *или* `switch` *със* `case` *вместо таблица за* δ *на следния автомат:*



Задача 2.8.2. *Да се направи програмна реализация и да се определи езика на следните автомати:*



2.9 Регулярни изрази и регулярни езици

Вече знаем как да превръщаме автоматите в реални програми. Това даде живот на нашата концептуална машина за валидация на текст. Ние не само имаме програмна реализация, тя също така има много хубави свойства. Въпреки всичко това има нещо което ни липсва. Понякога бихме искали да представим тази информация в по-компактен вариант от една картинка или програма. Тук идват на помощ регулярните изрази, които после ще покажем, че имат същата изразителна способност.

Дефиниция 2.9.1. Дефинираме множеството от **регулярни изрази** $\mathcal{RE}(\Sigma)$ индуктивно:

- $\varnothing, \varepsilon \in \mathcal{RE}(\Sigma)$ и $\sigma \in \mathcal{RE}(\Sigma)$ за всяко $\sigma \in \Sigma$
- ако $r_1, r_2 \in \mathcal{RE}(\Sigma)$, то тогава $r_1 \cdot r_2, r_1 + r_2 \in \mathcal{RE}(\Sigma)$
- ако $r \in \mathcal{RE}(\Sigma)$, то тогава $r^* \in \mathcal{RE}(\Sigma)$

Забележка. Освен това имаме и скоби за приоритет на операциите. При липса на скоби с най-голям приоритет е $*$, после $\cdot$, и накрая $+$. Също така за краткост понякога ще изпускаме $\cdot$ както се прави при умножението.

Дефиниция 2.9.2. Дефинираме индуктивно **езика** $\mathcal{L}[\![r]\!]$ **на регулярен израз** $r \in \mathcal{RE}(\Sigma)$:

- $\mathcal{L}[\![\varnothing]\!] = \varnothing$ и $\mathcal{L}[\![x]\!] = \{x\}$ за всяко $x \in \Sigma_\varepsilon$ $(\Sigma \cup \{\varepsilon\})$
- $\mathcal{L}[\![r_1 \cdot r_2]\!] = \mathcal{L}[\![r_1]\!] \cdot \mathcal{L}[\![r_2]\!]$, $\mathcal{L}[\![r_1 \cup r_2]\!] = \mathcal{L}[\![r_1]\!] \cup \mathcal{L}[\![r_2]\!]$, $\mathcal{L}[\![r^*]\!] = \mathcal{L}[\![r]\!]^*$

Дефиниция 2.9.3. Език $L \subseteq \Sigma^*$ наричаме **регулярен**, ако съществува регулярен израз $r \in \mathcal{RE}(\Sigma)$ такъв, че $\mathcal{L}[\![r]\!] = L$

Ето няколко прости примери за регулярни изрази и техните езици:

- $\mathcal{L}[\![(a+b)^*]\!] = \Sigma^*$
- $\mathcal{L}[\![((a+b)(a+b))^*]\!] = (\Sigma^2)^*$
- $\mathcal{L}[\![(b^*ab^*ab^*)^*]\!] = \{\alpha \in \Sigma^* \mid |\alpha| \equiv 0 \pmod 2\}$
- $\mathcal{L}[\![b^*ab^*a(b^*ab^*ab^*ab^*)^*]\!] = \{\alpha \in \Sigma^* \mid |\alpha| \equiv 2 \pmod 3\}$

Първите две са очевидни. Ще обосновем много накратко третия пример и ще оставим четвъртия на читателя. Едната посока е:

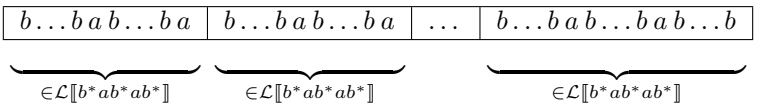
$$\mathcal{L}[\![(b^*ab^*ab^*)^*]\!] \subseteq \{\alpha \in \Sigma^* \mid |\alpha| \equiv 0 \pmod 2\}$$

Със $*$ наслагваме "блокчета" от $b^*ab^*ab^*$, които са все думи със две на брой букви a:

Колкото и пъти да събираме четни числа, винаги като резултат ще получим четно число. Думата очевидно е от $\{\alpha \in \Sigma^* \mid |\alpha| \equiv 0 \pmod 2\}$. Другата посока е:

$$\{\alpha \in \Sigma^* \mid |\alpha| \equiv 0 \pmod 2\} \subseteq \mathcal{L}[\![(b^*ab^*ab^*)^*]\!]$$

Ако вземем дума от ляво, можем да я разделим на всеки две срещания на буквата a:



Вече е очевидно, че думата е от $\mathcal{L}[\![(b^*ab^*ab^*)^*]\!]$.

2.10 Операции над регулярни езици

Сега ще разгледаме някои операции, които запазват регулярност.

Твърдение 2.10.1. *Ако L е регулярен език, то и езикът L^{rev} също е регулярен.*

Доказателство. С индукция по строене на регулярните езици.

- $L^{rev} = L$ за всеки $L \in \{\varnothing, \{\varepsilon\}, \{a\}, \{b\}\}$ ✓
- $(L_1 \cdot L_2)^{rev} = L_2^{rev} \cdot L_1^{rev}$ (от свойства на reverse)
  По ИП L_1^{rev} и L_2^{rev} са регулярни, откъдето $L_2^{rev} \cdot L_1^{rev}$ е регулярен.
- $(L_1 \cup L_2)^{rev} = L_1^{rev} \cup L_2^{rev}$ (директна проверка с дефиниции)
  По ИП L_1^{rev} и L_2^{rev} са регулярни, откъдето $L_2^{rev} \cup L_1^{rev}$ е регулярен.
- $(L^*)^{rev} = (L^{rev})^*$ (обобщение на предните две)
  По ИП L^{rev} е регулярен, откъдето $(L^{rev})^*$ е регулярен.

□

Дефиниция 2.10.2. За всяко $\alpha \in \Sigma^*$ дефинираме $sub(\alpha) = \{\beta \in \Sigma^* \mid \beta$ е подредица на $\alpha\}$

Твърдение 2.10.3. *Ако L е регулярен език, то и $sub[L]$ е регулярен.*

Доказателство. Ще покажем само структурата на индукцията.

- базата е очевидна ✓
- $sub[L_1 \cdot L_2] = sub[L_1] \cdot sub[L_2]$ (понеже $sub(\alpha \cdot \beta) = sub(\alpha) \cdot sub(\beta)$)
- $sub[L_1 \cup L_2] = sub[L_1] \cup sub[L_2]$ (така работят функциите)
- $sub[L^*] = (sub[L])^*$ (обобщение на предните две)

Остава само да се приложат индукционните предположения. □

2.11 Еквивалентност на регулярните и автоматните езици

Теорема 2.11.1 (Теорема на Клини). *За всеки език $L \subseteq \Sigma^*$:*

$$L \text{ е автоматен} \iff L \text{ е регулярен}$$

Доказателство. ($\Leftarrow$) С индукция по строене на регулярните езици.

- $\varnothing, \varepsilon, \{a\}, \{b\}$ са автоматни езици
- $\cup$, $\cdot$ и $*$ запазват автоматност

($\Rightarrow$) Тук ще трябва да поработим малко повече.
Нека $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, F \rangle$ е ДКА за L. Нека $Q = \{q_0, \ldots, q_{n-1}\}$. Със $L(i,j,k)$ ще бележим множеството от всички думи α, за които $\delta^*(q_i, \alpha) = q_j$ и всички междинни състояния имат индекс $< k$. Ясно е, че $L = \bigcup\{L(0,j,n) \mid q_j \in F\}$.
Строим $L(i,j,k)$ рекурсивно:
За $k = 0$ има две възможности (които тривиално са регулярни езици):

- за $i = j$ имаме, че $L(i,j,0) = \{\varepsilon\} \cup \{x \in \Sigma \mid \delta(q_i, x) = q_j\}$
- за $i \neq j$ имаме, че $L(i,j,0) = \{x \in \Sigma \mid \delta(q_i, x) = q_j\}$

Имайки $L(i,j,k)$, за $L(i,j,k+1)$ имаме две възможности:

- q_k да не се среща като междинно. Тогава сме в $L(i,j,k)$
- q_k се среща като междинно. Разделяме по срещанията на q_k:

$\in L(i,k,k)$	$\in L(k,k,k)$	$\ldots$	$\in L(k,k,k)$	$\in L(k,j,k)$
1-во срещане	2-ро срещане		последно срещане	

Тъй като го разделихме на всички срещания няма как да получим междинни с индекс k. Накрая:

$$L(i,j,k+1) = L(i,j,k) \cup (L(i,k,k) \cdot L(k,k,k)^* \cdot L(k,j,k))$$

□

Така добавихме още един начин за да опишем класът от езици, с който се занимаваме.

Внимание. От понятията регулярен език и автоматен език вече ще ползваме само регулярен.

2.12 Нерегулярни езици

До сега всичко, което сме правили е да се опитаме да докажем, че някакъв език е регулярен. Тогава идва естественият въпрос дали има нерегулярни езици, и ако да - как изглеждат те. Че има нерегулярни езици е ясно, иначе тази класификация на езици щеше да е безсмислена. Какъв обаче би бил един такъв език и как точно да докажем, че не може да се направи автомат или регулярен израз за него. Като че ли изглежда по-лесно да докажеш съществуването на обект като го конструираш, отколкото да докажеш, че такъв не може да се построи.
Ще вземем два метода за доказването на нерегулярност на езици. Кое кога да се използва е въпрос на лично предпочитание. За първият метод ще трябва да въведем една допълнителна дефиниция.

Дефиниция 2.12.1. Нека $\alpha \in \Sigma^*$ и $L \subseteq \Sigma^*$. Дефинираме $\alpha^{-1}(L) = \{\beta \in \Sigma^* \mid \alpha \cdot \beta \in L\}$

Твърдение 2.12.2. *Нека L е регулярен. Тогава множеството $\{\alpha^{-1}(L) \mid \alpha \in \Sigma^*\}$ е крайно.*

Доказателство. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е ДКА за L. Тогава:

$$\beta \in \alpha^{-1}(L) \iff \alpha \cdot \beta \in L \iff \delta^*(s, \alpha \cdot \beta) \in F \iff \delta^*(\delta^*(s,\alpha),\beta) \in F \iff \beta \in \mathcal{L}(\langle \Sigma, Q, \delta^*(s,\alpha), \delta, F \rangle)$$

От тук можем да видим, че ако L е регулярен, то за всяка дума α, на $\alpha^{-1}(L)$ съпоставяме състояние от Q т.е.

$$|\{\alpha^{-1}(L) \mid \alpha \in \Sigma^*\}| \leq |Q| < \infty$$

□

Следствие 2.12.3 (Критерий за нерегулярност)**.** *Ако* $\{\alpha^{-1}(L) \mid \alpha \in \Sigma^*\}$ *е безкрайно, то* L *не е регулярен.*

Доказателство. Контрапозиция на Твърдение 2.12.2 □

Започваме с каноничния пример за нерегулярен език:

Твърдение 2.12.4. *Езикът* $L = \{a^n b^n \mid n \in \mathbb{N}\}$ *не е регулярен.*

Доказателство. Искаме да покажем, че $\{\alpha^{-1}(L) \mid \alpha \in \Sigma^*\}$ е безкрайно. За целта ни трябват изброима редица от думи $\alpha_0, \alpha_1, \alpha_2, \ldots$ такава, че за $i \neq j : \alpha_i^{-1}(L) \neq \alpha_j^{-1}(L)$. Твърдя, че думите от вида a^n за $n \in \mathbb{N}$ ще ни свършат работа. Нека $n, k \in \mathbb{N}$, $n \neq k$. Тогава:

- $a^n b^n \in L \Rightarrow b^n \in (a^n)^{-1}(L)$
- $a^k b^n \notin L \Rightarrow b^n \notin (a^k)^{-1}(L)$

Така получаваме, че $(a^n)^{-1}(L) \neq (a^k)^{-1}(L)$.
Тук $(a^0)^{-1}(L), (a^1)^{-1}(L), (a^2)^{-1}(L), \ldots$ са безброй много съществено различни елементи на $\{\alpha^{-1}(L) \mid \alpha \in \Sigma^*\}$. Така по Критерий за нерегулярност езикът L не е регулярен. □

Задача 2.12.5. *Да се докаже, че следните езици не са регулярни:*

- $L_1 = \{a^n b^{2n} \mid n \in \mathbb{N}\}$
- $L_2 = \{a^{3n} b^n \mid n \in \mathbb{N}\}$
- $L_3 = \{a^{5n} b^{7n} \mid n \in \mathbb{N}\}$
- $L_4 = \{a^n b^{n^2} \mid n \in \mathbb{N}\}$
- $L_5 = \{a^n b^{2^n} \mid n \in \mathbb{N}\}$

Упътване: напълно аналогично на горния пример

Твърдение 2.12.6. *Езикът* $L = \{a^n b^m \mid n \leq m\}$ *не е регулярен.*

Доказателство. Отново ще пробваме с думи от вида a^n за $n \in \mathbb{N}$. Нека $n, k \in \mathbb{N}$, $n \neq k$ като Б.О.О. $n < k$.

- Понеже $n \leq n$, $a^n b^n \in L$, откъдето $b^n \in (a^n)^{-1}(L)$
- Понеже $n < k \iff \neg(k \leq n)$, $a^k b^n \notin L$, откъдето $b^n \notin (a^k)^{-1}(L)$

По Критерий за нерегулярност езикът L не е регулярен. □

Твърдение 2.12.7. *Езикът* $L = \{a^{n^2} \mid n \in \mathbb{N}\}$ *не е регулярен.*

Доказателство. Отново ще пробваме с думи от вида a^n за $n \in \mathbb{N}$. Нека $n, k \in \mathbb{N}$, $n \neq k$ като Б.О.О. $k < n$.

- $a^n a^{n^2+n+1} = a^{(n+1)^2} \in L$, откъдето $a^{n^2+n+1} \in (a^n)^{-1}(L)$
- $n^2 < n^2 + n + k + 1 < n^2 + 2n + 1 = (n+1)^2$, следователно $a^k a^{n^2+n+1} \notin L$, откъдето $a^{n^2+n+1} \notin (a^k)^{-1}(L)$

По Критерий за нерегулярност езикът L не е регулярен. □

Това е често срещана техника при езици от вида $\{a^{f(n)} \mid n \in \mathbb{N}\}$, за някоя $f : \mathbb{N} \to \mathbb{N}$ строго растяща. За да се "изкара" думата от езика е много удобно да се използва, че ако $f(n) < x < f(n+1)$, то x няма как да е член на редицата.

Задача 2.12.8. *Да се докаже, че следните езици не са регулярни:*

- $L_1 = \{a^{n^3} \mid n \in \mathbb{N}\}$
- $L_2 = \{a^{2^n} \mid n \in \mathbb{N}\}$
- $L_3 = \{a^{n!} \mid n \in \mathbb{N}\}$
- $L_4 = \{a^{1+\cdots+n} \mid n \in \mathbb{N}\}$

Упътване: да се използва горепоказаната техника

Твърдение 2.12.9. *Езикът* $L = \{\alpha\alpha^{rev} \mid \alpha \in \Sigma^*\}$ *е нерегулярен.*

Доказателство. Тук думи от вида a^n няма да ни свършат работа. Изобщо при еднобуквена азбука езикът е регулярен. Получават се думи с четна дължина. Ще трябва да използваме и други букви за да успеем да изкараме този резултат. Нека $n, k \in \mathbb{N}$, $n \neq k$.

- $a^n b^n b^n a^n \in L$
- $a^k b^n b^n a^n \notin L$ (очевидно $a^k \neq a^n$, а $a^k b^t$ за $0 \leq t \leq 2n$ са единствените кандидати за α, че да стане $\alpha\alpha^{rev}$)

По Критерий за нерегулярност езикът L не е регулярен. □

Задача 2.12.10. *Да се докаже, че следните езици не са регулярни:*

- $L_1 = \{\alpha\alpha \mid \alpha \in \Sigma^*\}$
- $L_2 = \{\alpha\alpha\alpha \mid \alpha \in \Sigma^*\}$
- $L_3 = \{\alpha\alpha^{rev}\alpha \mid \alpha \in \Sigma^*\}$

Упътване: напълно аналогично на горния пример

Нека сега видим какво се случва с различните операции. Ако L не е регулярен, то и $\overline{L}$ също не е регулярен. В противен случай $\overline{\overline{L}} = L$ би бил регулярен. Същите разсъждения можем да направим за rev. Тази затвореност я няма при други операции, които сме разглеждали. Тук се запазва поради обратимостта на тази операция.

Внимание. Следните са често срещани грешки:

- ***конкатенация на два нерегулярни езика винаги ни дава нерегулярен език***
  Нека вземем за пример някой нерегулярен език L. Тогава $\overline{L}$ също няма да е регулярен. Добавяйки крайно много елементи към който и да е нерегулярен език го оставя нерегулярен. В противен случай бихме махнали тези крайно много думи (те образуват регулярен език) и ще получим, че първоначалният език е регулярен. Въпреки че $(L \cup \{\varepsilon\})$ и $(\overline{L} \cup \{\varepsilon\})$ не са регулярни езици, конкатенацията им $(L \cup \{\varepsilon\}) \cdot (\overline{L} \cup \{\varepsilon\}) = \Sigma^*$ е регулярен език. Също така не е вярно, че при конкатенация на нерегулярни се получава винаги регулярен. Тривиално се проверява, че $\{a^n b^n \mid n \in \mathbb{N}\} \cdot \{a^n b^n \mid n \in \mathbb{N}\} = \{a^n b^n a^m b^m \mid n, m \in \mathbb{N}\}$ не е регулярен.
- ***обединение на два нерегулярни езика винаги ни дава нерегулярен език***
  Абсолютно същите примери вършат работа и тук. За произволен L, $L \cup \overline{L} = \Sigma^*$ е регулярен. Обратното също не е вярно, защото за нерегулярен език L, $L \cup L = L$ е нерегулярен.
- ***подмножество на регулярен/нерегулярен език е регулярен/нерегулярен***
  $\varnothing$ е подмножество на всеки език, а Σ^* разширява всеки език.

2.13 Лема за покачването

Сега ще покажем другия критерий за доказване на нерегулярност. Той понякога е по-удобен за използване, обаче не е винаги приложим, и е по-вербозен.

Лема 2.13.1 (Лема за покачването)**.** *Нека* L *е регулярен език. Тогава:*

$$(\exists p \geq 1)$$
$$(\forall \alpha \in L,\ |\alpha| \geq p)$$
$$(\exists x, y, z \in \Sigma^*,\ xyz = \alpha,\ |xy| \leq p,\ |y| \geq 1)$$
$$(\forall i \in \mathbb{N})[xy^i z \in L]$$

Доказателство. Езикът L е регулярен. Следователно съществува ДКА $\mathcal{A} = \langle Q, \Sigma, s, \delta, F \rangle$, такъв че $\mathcal{L}(\mathcal{A}) = L$.
Полагаме $p = |Q|$ и нека $q_1, \ldots, q_p$ са състоянията от Q.
Нека $\alpha \in L$ е такава, че $|\alpha| = n$, където $n \geq p$. Ще разбием α на $\alpha_1, \ldots, \alpha_n \in \Sigma$ (т.е. $\alpha = \alpha_1 \ldots \alpha_n$).
Знаем, че съществуват $i_0, \ldots, i_n \in \{1, \ldots, n\}$ такива, че $s = q_{i_0}$ и за всяко $j \in \{1, \ldots, n\} : \delta(q_{i_{j-1}}, \alpha_j) = q_{i_j}$.
Нека разгледаме думата $\alpha_1 \ldots \alpha_p$. За нея знаем, че по време на четенето на думата автоматът минава през $p+1$ състояния. Следователно по принципът на Дирихле съществуват $t_1, t_2 \in \{1, \ldots, p\}$, където $t_1 < t_2$ такива, че $q_{i_{t_1}} = q_{i_{t_2}}$.
Полагаме $x = \alpha_1 \ldots \alpha_{t_1}$, $y = \alpha_{t_1+1} \ldots \alpha_{t_2}$ и $z = \alpha_{t_2+1} \ldots \alpha_n$. Сигурни сме, че $|xy| \leq p$, защото $t_2 \leq p$ и че $|y| \geq 1$ понеже $t_1 \neq t_2$. Знаем, че $\delta^*(q_{i_{t_1}}, y) = q_{i_{t_2}}$. Искаме да проверим, че можем да "циклим" със y.

Твърдение 2.13.2. $(\forall i \in \mathbb{N})(\delta^*(q_{i_{t_1}}, y^i) = q_{i_{t_2}})$.

Доказателство. Ще докажем твърдението с индукция по $i \in \mathbb{N}$.
База: $\delta^*(q_{i_{t_1}}, \epsilon) = q_{i_{t_1}} = q_{i_{t_2}}$ ✓
ИС: $\delta^*(q_{i_{t_1}}, y^{i+1}) = \delta(\delta^*(q_{i_{t_1}}, y^i), y) \overset{\text{ИП}}{=} \delta(q_{i_{t_2}}, y) = \delta(q_{i_{t_1}}, y) = q_{i_{t_2}}$ □

Знаем, че $\alpha = xyz \in L$. Тогава $\delta^*(s, xyz) = \delta^*(\delta^*(s, xy), z) = \delta^*(\delta^*(\delta^*(s, x), y), z) \in F$.
Така получаваме, че $\delta^*(s, x) = q_{i_{t_1}}$ и от там $\delta^*(\delta^*(q_{i_{t_1}}, y), z) \in F$. От Твърдение 2.13.2 имаме, че за всяко $i \in \mathbb{N}$, $\delta^*(q_{i_{t_1}}, y^i) = q_{i_{t_2}} = q_{i_{t_1}}$. Освен това знаем, че $\delta^*(q_{i_{t_2}}, z) \in F$.
Следователно за всяко $i \in \mathbb{N}$, $\delta^*(\delta^*(q_{i_{t_1}}, y^i), z) \in F$. От тук можем да заключим, че $\delta^*(\delta^*(\delta^*(s, x), y^i), z) \in F$ за всяко $i \in \mathbb{N}$. и вървейки в обратната посока ("сливането" на всички δ^* в едно) получаваме, че за всяко $i \in \mathbb{N}$, $\delta^*(s, xy^iz) \in F$, с което доказахме лемата. □

Тук отново няма да използваме твърдението в тази форма, а неговата контрапозиция:

Следствие 2.13.3 (Лема за покачването (контрапозиция))**.** *Ако е изпълнено, че:*

$$(\forall p \geq 1)$$
$$(\exists \alpha \in L,\ |\alpha| \geq p)$$
$$(\forall x, y, z \in \Sigma^*,\ xyz = \alpha,\ |xy| \leq p,\ |y| \geq 1)$$
$$(\exists i \in \mathbb{N})[xy^iz \notin L],$$

то тогава L не е регулярен.

Ще покажем и по този начин, че $L = \{a^nb^n \mid n \in \mathbb{N}\}$ не е регулярен. Нека $p \geq 1$. Тогава $\alpha = a^pb^p \in L$, $|a^pb^p| = 2p \geq p$. Нека вземем $x, y, z \in \Sigma^*$, $xyz = \alpha$, $|xy| \leq n$, $|y| \geq 1$. Знаем със сигурност, че $y = a^t$ за някое $1 \leq t \leq n$. За да намерим числото $i \in \mathbb{N}$, което ще ни "изкара" думата, трябва да видим как изглежда xy^iz:

$$xy^iz = a^pa^{(i-1)t}b^p$$

Тук всяко $i \neq 1$ ще ни свърши работа. Нека вземем $i = 0$. Тогава $p + (i-1)t = p - t < p$, понеже $(t \geq 1)$, откъдето $xy^0z = a^{(p-t)}b^p \notin L$. Така по Лема за покачването (контрапозиция) излиза, че езикът не е регулярен.

Твърдение 2.13.4. *Езикът $L = \{a^p \mid p$ е просто число$\}$ не е регулярен.*

Доказателство. Нека $n \geq 1$ и $p \geq n$ е просто число.
Тогава $\alpha = a^p \in L$, $|a^p| \geq n$. Нека $x, y, z \in \Sigma^*$, $xyz = \alpha$, $|xy| \leq n$, $|y| \geq 1$. Тогава $y = a^t$ за някое $1 \leq t \leq n$. Нека видим как изглежда xy^iz:

$$xy^iz = xyy^{i-1}z = a^pa^{(i-1)t}$$

Искаме $p + (i-1)t$ да е съставно число. t е нещо, което се променя, така че по-скоро бихме се надявали p да е множител. Искаме да е множител във $(i-1)t$. Възможно е $t < p$, така че трябва $p \mid i - 1$. Ако положим $i = p + 1$ получаваме, че:

$$p + (i-1)t = p + (p + 1 - 1)t = p + pt = p(1 + t)$$

Тъй като $t \geq 1$, $t + 1 \geq 2$, и от там $p(1 + t) = p + (i-1)t$ е съставно число. Накрая понеже $xy^{p+1}z = a^{p(1+t)} \notin L$, по Лема за покачването (контрапозиция) излиза, че L не е регулярен. □

Задача 2.13.5. *С лемата за покачването да се докаже, че следните езици не са регулярни:*

- $L_1 = \{a^nb^n \mid n \in \mathbb{N}\}$ *// у ще хване само букви а*
- $L_2 = \{a^{n^2} \mid n \in \mathbb{N}\}$
- $L_3 = \{a^{2^n} \mid n \in \mathbb{N}\}$
- $L_4 = \{a^{n!} \mid n \in \mathbb{N}\}$
- $L_5 = \{\alpha\alpha \mid \alpha \in \Sigma^*\}$
- $L_6 = \{\alpha\alpha^{rev} \mid \alpha \in \Sigma^*\}$

Упътване: решенията на тази задача са просто адаптация на тези от Задача 2.12.5, Задача 2.12.8 и Задача 2.12.10

От всичкото това доказване на нерегулярност използвайки лемата, човек си задава въпроса дали това не може да стане и в обратната посока. Оказва се, че не може.

Внимание. Има **нерегулярни** езици, за който условието от Лема за покачването **е изпълнено**.

За пример нека вземем:

$$L = (\{c\}^+ \cdot \{a^n b^n \mid n \in \mathbb{N}\}) \cup (\{a\}^* \cdot \{b\}^*)$$

Ако допуснем, че L е регулярен, то тогава очевидно:

$$L \cap (\{c\} \cdot \{a\}^* \cdot \{b\}^*) = \{c\} \cdot \{a^n b^n \mid n \in \mathbb{N}\}$$

ще бъде регулярен, но той не е (елементарна проверка, която оставяме на читателя). Сега да проверим условието от Лема за покачването. Нека $p = 2$. Нека $\alpha \in L$, $|\alpha| \geq p$. Полагаме:

- $x = \varepsilon$
- $y = \alpha[1]$ (първата буква на α)
- $z = \alpha[2 : |\alpha|]$ (останалите букви на α)

Ако $\alpha \in \{c\}^+ \cdot \{a^n b^n \mid n \in \mathbb{N}\}$, то тогава $y = c$. За $i = 2$, $xy^i z = c \cdot \alpha \in L$. Ако пък $\alpha \in \{a\}^* \cdot \{b\}^*$, то тогава очевидно $i = 2$ пак ще свърши работа.

2.14 Автомат на Brzozowski

Автоматите, които създаваме, не са единствените за конкретния език. Най-малкото можем да добавим недостижими състояния. Но това е глупаво, ние по-скоро искаме да направим автомат с възможно най-малко състояния. Тогава ако решим да направим програма, тя ще заема по-малко памет.

Дефиниция 2.14.1. За език $L \subseteq \Sigma^*$ дефинираме $Q_L = \{\alpha^{-1}(L) \mid \alpha \in \Sigma^*\}$.

Твърдение 2.14.2. *Ако Q_L е крайно, то L е регулярен.*

Доказателство. Нека Q_L е крайно.
Строим автомат $\mathcal{B}_L = \langle \Sigma, Q_L, L, \delta, F \rangle$ за L, който ще наричаме **автомат на Brzozowski** за L:

- $\delta(M, x) = x^{-1}(M)$ за $M \in Q_L$, $x \in \Sigma$
- $F = \{M \in Q_L \mid \varepsilon \in M\}$

Твърдение 2.14.3. *За всяко $M \in Q_L$: $\delta^*(M, \alpha) = \alpha^{-1}(M)$*

Доказателство. С индукция по $|\alpha|$.

- $\delta^*(M, \varepsilon) = M = \varepsilon^{-1}(M)$ ✓
- $\delta^*(M, \beta x) = \delta(\delta^*(M, \beta), x) \overset{\text{ИП}}{=} x^{-1}(\beta^{-1}(M)) = \{\gamma \in \Sigma^* \mid x\gamma \in \beta^{-1}(M)\} = \{\gamma \in \Sigma^* \mid \beta x \gamma \in M\} = (\beta x)^{-1}(M)$

□

Имайки това:

$$\begin{aligned} \alpha \in L &\iff \varepsilon \in \alpha^{-1}(L) \\ &\iff \varepsilon \in \delta^*(L, \alpha) \\ &\iff \delta^*(L, \alpha) \in F \\ &\iff \alpha \in \mathcal{L}(\mathcal{B}) \end{aligned}$$

□

Вече сме сигурни, че Критерий за нерегулярност е напълно точен, за разлика от Лема за покачването:

Следствие 2.14.4. *L е регулярен $\iff$ Q_L е крайно*

Освен че за всеки регулярен език L може да се построи такъв автомат, той се оказва и минимален.

Твърдение 2.14.5 (Автоматът на Brzozowski е минимален)**.** *Нека L е регулярен с автомат $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$. Тогава $|Q_L| \leq |Q|$.*

Доказателство. Б.О.О. всички състояния в $\mathcal{A}$ са достижими.
Нека $Q = \{1, \ldots, n\}$. Нека $\alpha_i \in \Sigma^*$ е такава дума, че $\delta^*(s, \alpha) = q_i$. Дефинираме функцията $f : Q \to Q_L$:

$$f(q_i) = \alpha^{-1}(L)$$

Трябва само да покажем, че f е сюрективна.
Нека $M \in Q_L$. Тогава има $\alpha \in \Sigma^* \ : \ M = \alpha^{-1}(L)$. Знаем, че $\delta^*(s, \alpha) = q_i$ за някое $i \in \{1, \ldots, n\}$.

$$\begin{aligned} \beta \in M \iff \beta \in \alpha^{-1}(L) &\iff \alpha\beta \in L \\ &\iff \delta^*(s, \alpha\beta) \in F \iff \delta^*(\delta^*(s, \alpha), \beta) \in F \iff \delta^*(q_i, \beta) \in F \\ &\iff \delta^*(\delta^*(s, \alpha_i), \beta) \in F \iff \delta^*(s, \alpha_i\beta) \in F \iff \alpha_i\beta \in L \\ &\iff \beta \in \alpha_i^{-1}(L) \iff \beta \in f(q_i) \end{aligned}$$

Така $f(q_i) = M$. Накрая понеже f е сюрективна, $\mathrm{Dom}(f) = Q$, $\mathrm{Rng}(f) = Q_L$, $|Q_L| \leq |Q|$. □

Задача 2.14.6. *Да се построи минимален автомат за $\mathcal{L}[\![a^*b^*]\!]$*

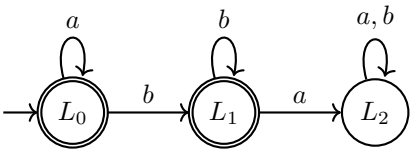
За такива задачи е хубаво да се имат на предвид следните свойства на регулярните изрази:

- $\mathcal{L}[\![r^+]\!] = \mathcal{L}[\![r \cdot r^*]\!]$
- $\mathcal{L}[\![r^*]\!] = \mathcal{L}[\![r^+ + \varepsilon]\!]$
- $\mathcal{L}[\![r_1 + r_2]\!] = \mathcal{L}[\![r_2 + r_1]\!]$
- $\mathcal{L}[\![r(r_1 + r_2)]\!] = \mathcal{L}[\![r \cdot r_1 + r \cdot r_2]\!]$

Започваме да строим автомат на Brzozowski за $\mathcal{L}[\![a^*b^*]\!]$:

- начално състояние $L_0 = \mathcal{L}[\![a^*b^*]\!] = \mathcal{L}[\![a^+b^* + b^*]\!] = \mathcal{L}[\![a \cdot a^*b^* + b^*]\!] = \mathcal{L}[\![a \cdot a^*b^* + b^+ + \varepsilon]\!] = \mathcal{L}[\![a \cdot a^*b^* + b \cdot b^* + \varepsilon]\!]$
- функцията на преходите е следната:
  - $\delta(L_0, a) = a^{-1}(\mathcal{L}[\![a \cdot a^*b^* + b \cdot b^* + \varepsilon]\!]) = \mathcal{L}[\![a^*b^*]\!] = L_0$
  - $\delta(L_0, b) = b^{-1}(\mathcal{L}[\![a \cdot a^*b^* + b \cdot b^* + \varepsilon]\!]) = \mathcal{L}[\![b^*]\!] = \mathcal{L}[\![b \cdot b^* + \varepsilon]\!] = L_1 \neq L_0$, понеже $a \in \mathcal{L}[\![a^*b^*]\!]$, $a \notin \mathcal{L}[\![b^*]\!]$
  - $\delta(L_1, a) = a^{-1}(\mathcal{L}[\![b \cdot b^* + \varepsilon]\!]) = \varnothing = L_2 \neq L_0, L_1$
  - $\delta(L_1, b) = b^{-1}(\mathcal{L}[\![b \cdot b^* + \varepsilon]\!]) = \mathcal{L}[\![b^*]\!] = L_1$
  - $\delta(L_2, a) = a^{-1}(\varnothing) = \varnothing = b^{-1}(\varnothing) = \delta(L_2, a)$
- $\varepsilon \in L_0, L_1$, следователно те стават финални

Ето как ще изглежда автомата на картинка:



минимален автомат за $\mathcal{L}[\![a^*b^*]\!]$

Как бихме процедирали ако трябва да се построи минимален автомат за $\Sigma^* \setminus \mathcal{L}[\![a^*b^*]\!]$?

Внимание. За всеки регулярен език L, ако има автомат за L с n на брой състояния, то тогава има автомат за $\overline{L}$ с n на брой състояния.

Това го знаем още от Твърдение 2.4.1. В конструкцията там ние използваме същото множество от състояния, същото начално състояния и същите преходи. Единственото, което променяме, е кои са финалните състояния. Така можем да заключим, че за да получим минималният автомат за $\Sigma^* \setminus \mathcal{L}[\![a^*b^*]\!]$, трябва просто да приложим конструкцията от Твърдение 2.4.1 върху минималният автомат за $\mathcal{L}[\![a^*b^*]\!]$. Това е много по-кратко от преминаването към регулярен израз за $\Sigma^* \setminus \mathcal{L}[\![a^*b^*]\!]$ и прилагането на алгоритъма на Brzozowski.

2.15 Задачи за упражнение

Дефиниция 2.15.1. Функция $h : \Sigma_1^* \to \Sigma_2^*$ ще наричаме **хомоморфизъм**, ако:

$$h(\alpha \cdot \beta) = h(\alpha) \cdot h(\beta) \text{ за всички } \alpha, \beta \in \Sigma_1^*$$

Задача 2.15.2. *Да се докаже, че за произволен хомоморфизъм h е вярно, че $h(\varepsilon) = \varepsilon$.*

Задача 2.15.3. *Да се докаже, че за произволен хомоморфизъм h и регулярен език L, езикът $h[L]$ е регулярен.*
Упътване: да се направи индукция по строенето на регулярните езици

Задача 2.15.4. *Да се докаже, че за произволен хомоморфизъм h и регулярен език L, езикът $h^{-1}[L]$ е регулярен.*
Упътване: да се направи автомат, който докато чете α, прави преходи с $h(\alpha)$ в автомат за L

Задача 2.15.5. *Използвайки нерегулярността на $\{a^n b^n \mid n \in \mathbb{N}\}$ заедно със Задача 2.15.3 или Задача 2.15.4 да се докаже, че езикът $L = \{a^n b^{2n} \mid n \in \mathbb{N}\}$ е нерегулярен.*
Упътване: да се представи L като образ или праобраз на хомоморфизъм

Задача 2.15.6. *Нека L е регулярен език. Да се докаже, че езикът $L' = \{\alpha \in \Sigma^* \mid \alpha\alpha \in L\}$ е регулярен.*
Упътване: докато се чете α да се вижда къде отиват всички състояния от автомата за L

Задача 2.15.7. *Нека L е регулярен език. Да се докаже, че езикът $L' = \{\alpha \# c^n \mid n \in \mathbb{N} \,\&\, \alpha^n \in L\}$ е регулярен.*
Упътване: леко усложнение на конструкцията от предната задача

Задача 2.15.8. *Нека L е регулярен език. Да се докаже, че езикът $L' = \{\alpha \in \Sigma^* \mid (\exists n \in \mathbb{N})\,(\alpha^n \in L)\}$ е регулярен.*
Упътване: да се използват предната задача и хомоморфизми

Дефиниция 2.15.9. Функция $h : \Sigma_1^* \to \mathcal{P}(\Sigma_2^*)$ ще наричаме **регулярен хомоморфизъм**, ако:

- $h(x)$ е регулярен за всяко $x \in \Sigma_1$
- $h(\alpha \cdot \beta) = h(\alpha) \cdot h(\beta)$ за всички $\alpha, \beta \in \Sigma_1^*$

Задача 2.15.10. *Да се докаже, че за произволен регулярен хомоморфизъм h е вярно, че $h(\varepsilon) = \{\varepsilon\}$*

Задача 2.15.11. *Да се докаже, че за произволен регулярен хомоморфизъм h и регулярен език L, $\bigcup h[L]$ е регулярен.*
Упътване: да се направи индукция по строенето на регулярните езици

ГЛАВА

3

ГРАМАТИКИ И СТЕКОВИ АВТОМАТИ

Сега ще покажем едно много по-мощно средство, което може да разпознава много повече езици като цената, която ще платим, е да се загубят няколко хубави свойства.

3.1 Безконтекстни граматики

Дефиниция 3.1.1. Безконтекстна граматика ще наричаме всяко $G = \langle \Sigma, V, S, R \rangle$, където:

- Σ е крайна азбука
- V е крайно множество от променливи
- $S \in V$ (ще го наричаме начална променлива)
- $R \subseteq V \times (\Sigma \cup V)^*$ е крайно множество от правила

Забележка. Когато $\langle A, \alpha \rangle \in R$ ще го бележим с $A \to_G \alpha$. Ако граматиката G се подразбира, може да пишем $A \to \alpha$.

За правилата $A \to \alpha$ можем да си мислим, че означават A се замения с α. Искаме да видим какви думи могат да се генерират започвайки от S и следвайки правилата R.

Дефиниция 3.1.2. За произволна безконтекстна граматика $G = \langle \Sigma, V, S, R \rangle$ и $l \in \mathbb{N}$ дефинираме релацията $\overset{l}{\lhd}_G \subseteq (\Sigma \cup V) \times (\Sigma \cup V)^*$ индуктивно:

- $X \overset{0}{\lhd}_G X$ за всяко $X \in \Sigma \cup V$
- Ако $X \in V$, $X_i \in \Sigma \cup V$, в G имаме правилото $X \to X_1 \dots X_n$, и $X_i \overset{l_i}{\lhd}_G \alpha_i$ за някои $l_i \in \mathbb{N}$ и $\alpha_i \in (\Sigma \cup V)^*$, то $X \overset{l+1}{\lhd}_G \alpha_1 \dots \alpha_n$, където $l = \max\{l_1, \dots, l_n\}$. Тук включваме случая с ε. Тъй като $\max(\varnothing) = 0$, $X \overset{1}{\lhd}_G \varepsilon$.

Пишем $X \overset{*}{\lhd}_G \alpha$, ако има $l \in \mathbb{N}$ такова, че $X \overset{l}{\lhd}_G \alpha$. Ако G се подразбира, не го пишем.

Дефиниция 3.1.3. Нека $G = \langle \Sigma, V, S, R \rangle$ безконтекстна граматика и $A \in V$. Тогава езикът на променливата A ще бъде $\mathcal{L}_G(A) = \{\alpha \in \Sigma^* \mid A \overset{*}{\lhd} \alpha\}$ и езикът на граматиката G ще бъде $\mathcal{L}(G) = \mathcal{L}_G(S)$.
Един език L наричаме **безконтекстен**, ако съществува безконтекстна граматика G такава, че $\mathcal{L}(G) = L$.

Забележка. За да пестим писане, когато имаме правилата $A \to \alpha_i$ ($i \in \{1, \dots, n\}$), ще записваме $A \to \alpha_1 \mid \dots \mid \alpha_n$. Ясно е, че като пишем граматика е напълно достатъчно да опишем правилата и да споменем коя е началната променлива. Ако пък има само една променлива, даже и това не е нужно.

Вече сме готови да дадем първият пример за безконтекстен език.

Твърдение 3.1.4. *Езикът $L = \{a^n b^n \mid n \in \mathbb{N}\}$ е безконтекстен.*

Доказателство. Граматиката за L е изключително проста (само с 2 правила):

$$S \to aSb \mid \varepsilon$$

Искаме сега да докажем, че $\mathcal{L}(G) = L$.
В посоката $\mathcal{L}(G) \subseteq L$ ще трябва да покажем, че ако $S \overset{l}{\lhd} \alpha$ и $\alpha \in \Sigma^*$, то $\alpha \in L$. Правим индукция по l:

- $S \overset{0}{\lhd} S \notin \Sigma^*$ ✓
- Нека $S \overset{l+1}{\lhd} \alpha \in \Sigma^*$. Тогава сме приложили някое правило.

  1 сл. Приложили сме правилото $S \to \varepsilon$. Тогава $\alpha = \varepsilon = a^0 b^0 \in L$.

  2 сл. Приложили сме правилото $S \to aSb$. Тогава $S \overset{l}{\lhd} \beta$ за някое $\beta \in \Sigma^*$ и $\alpha = a\beta b$. По ИП $\beta \in L$, откъдето $\beta = a^n b^n$ за някое $n \in \mathbb{N}$. Така $\alpha = a\beta b = a \cdot a^n b^n \cdot b = a^{n+1} b^{n+1} \in L$.

Така ако $\alpha \in \mathcal{L}(G)$, то $S \overset{*}{\lhd} \alpha$ т.е. $S \overset{l}{\lhd} \alpha$ за някое $l \in \mathbb{N}$, откъдето $\alpha \in L$.
В посоката $L \subseteq \mathcal{L}(G)$ ще покажем, че за всяко $n \in \mathbb{N}$, $S \overset{n+1}{\lhd} a^n b^n$. Доказваме с индукция по n:

- $S \overset{0}{\lhd} S$ и $S \to \varepsilon$, откъдето $S \overset{1}{\lhd} \varepsilon = a^0 b^0$ ✓
- По ИП $S \overset{n+1}{\lhd} a^n b^n$, но освен това $S \to aSb$, откъдето $S \overset{n+2}{\lhd} a \cdot a^n b^n \cdot b = a^{n+1} b^{n+1}$.

Така ако $\alpha \in L$, то понеже $\alpha = a^n b^n$ за някое $n \in \mathbb{N}$, то тогава $\alpha \in \mathcal{L}(G)$. □

Задача 3.1.5. *Да се докаже, че следните езици са безконтекстни:*

- $L_1 = \{b^n a^n \mid n \in \mathbb{N}\}$
- $L_2 = \{b^{2n} a^n \mid n \in \mathbb{N}\}$

Упътване: да се направи дребна модификация на граматиката от Твърдение 3.1.4

Твърдение 3.1.6. *Езикът $L = \{\alpha\alpha^{rev} \mid \alpha \in \Sigma^*\}$ е безконтекстен.*

Доказателство. Граматиката е следната:

$$S \to aSa \mid bSb \mid \varepsilon$$

Ще покажем, че ако $S \overset{l}{\lhd} \beta$ и $\beta \in \Sigma^*$, то $\beta = \alpha\alpha^{rev}$ за някое $\alpha \in \Sigma^*$. Правим индукция по l:

- $S \overset{0}{\lhd} S \notin \Sigma^*$ ✓
- Нека $S \overset{l+1}{\lhd} \beta \in \Sigma^*$. Тогава сме приложили някое правило.

  1 сл. Приложили сме правилото $S \to \varepsilon$. Тогава $\alpha = \varepsilon = \varepsilon\varepsilon^{rev}$.

  2 сл. Приложили сме правилото $S \to aSa$. Тогава $S \overset{l}{\lhd} \gamma$ за някое $\gamma \in \Sigma^*$ и $\beta = a\gamma a$. По ИП има $\alpha \in \Sigma^*$ такова, че $\gamma = \alpha\alpha^{rev}$. Така $\beta = a\gamma a = a\alpha\alpha^{rev}a = a\alpha(a\alpha)^{rev}$.

  3 сл. Приложили сме правилото $S \to bSb$. Той е аналогичен на 2 сл.

Така ако $\beta \in \mathcal{L}(G)$, $S \overset{*}{\lhd} \beta$ т.е. $S \overset{l}{\lhd} \beta$ за някое $l \in \mathbb{N}$, следователно $\beta = \alpha\alpha^{rev}$ за някое $\alpha \in \Sigma^*$, откъдето $\beta \in L$.
Сега ще покажем с индукция по $|\alpha|$, че $S \overset{|\alpha|+1}{\lhd} \alpha\alpha^{rev}$:

- $S \overset{0}{\lhd} S$ и $S \to \varepsilon$, откъдето $S \overset{1}{\lhd} \varepsilon = \varepsilon\varepsilon^{rev}$ ✓
- Нека $\alpha = x\beta$. По ИП $S \overset{|\beta|+1}{\lhd} \beta\beta^{rev}$, освен това имаме правилото $S \to xSx$, откъдето $S \overset{|\beta|+2}{\lhd} x\beta\beta^{rev}x$

Така ако $\alpha \in L$, то $\alpha = \beta\beta^{rev}$ за някое $\beta \in \Sigma^*$, но за β знаем, че $S \overset{*}{\lhd} \beta\beta^{rev} = \alpha$ т.е. $\alpha \in \mathcal{L}(G)$. □

Задача 3.1.7. *Да се докаже, следните езици са безконтекстни:*

- $L_1 = \{\alpha \in \Sigma^* \mid \alpha$ *е палиндром с нечетна дължина*$\}$
- $L_2 = \{\alpha \in \Sigma^* \mid \alpha = \alpha^{rev}\}$

Упътване: да се модифицира граматиката от Твърдение 3.1.6.

За да свикнем повече с $\overset{*}{\lhd}$ и за да си улесним някои доказателства, ще покажем, че можем да "слепваме" изводи:

Твърдение 3.1.8. *Ако* $X \overset{l}{\lhd} X_1 \dots X_n$ *и* $X_i \overset{*}{\lhd} \alpha_i$ *за* $X_i \in \Sigma \cup V$, $\alpha_i \in (\Sigma \cup V)^*$, *то* $X \overset{*}{\lhd} \alpha_1 \dots \alpha_n$.

Доказателство. С индукция по l.

- Нека $X \overset{0}{\lhd} X$. Ако $X \overset{*}{\lhd} \alpha$, то очевидно $X \overset{*}{\lhd} \alpha$ ✓
- Нека $X \overset{l+1}{\lhd} X_1 \dots X_n$. Тогава има $0 = j_0 < j_1 < \dots < j_k = n$ такива, че $X \to X'_1 \dots X'_k$ и $X'_i \overset{l'_i}{\lhd} X_{j_{i-1}+1} \dots X_{j_i}$ като $l = \max\{l'_1, \dots, l'_n\}$. Ако $X_i \overset{l_i}{\lhd} \alpha_i$ за $i \in \{1, \dots, n\}$, то по ИП за $t \in \{1, \dots, k\}$, $X'_t \overset{l'_t + l'_{t,max}}{\lhd} \alpha_{j_{t-1}+1} \dots \alpha_{j_t}$, където $l'_{t,max} = \max\{l_{j_{t-1}+1}, \dots, l_{j_t}\}$. Тогава $X \overset{1+l_{max}}{\lhd} \alpha_1 \dots \alpha_n$, където $l_{max} = \max\{l'_1 + l'_{1,max}, \dots, l'_k + l'_{k,max}\}$.

□

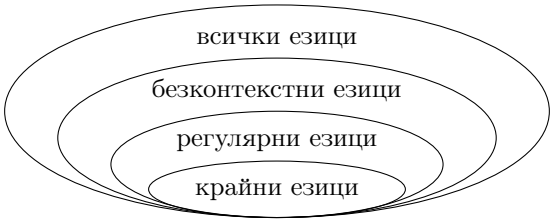
Твърдение 3.1.9. *Всеки регулярен език е безконтекстен.*

Доказателство. Ясно е, че $\varnothing, \{\varepsilon\}, \{a\}, \{b\}$ са безконтекстни. Остава да покажем, че безконтекстните езици са затворени относно регулярните операции.
Нека $G_i = \langle \Sigma, V_i, S_i, R_i \rangle$ са безконтекстни граматики за $i = 1, 2$. Нека Б.О.О. $V_1 \cap V_2 = \varnothing$. Нека $S \notin V_1 \cup V_2$. Следните твърдения са верни:

- $S \to_G S_1 \mid S_2$ заедно с правилата R_1, R_2 ще генерира езикът $\mathcal{L}(G_1) \cup \mathcal{L}(G_2)$.
  Ако $S \overset{*}{\lhd_G} \alpha$ и $\alpha \in \Sigma^*$, то тогава очевидно понеже $S \to_G S_i$ $(i = 1, 2)$ са единствените правила, $S_1 \overset{*}{\lhd_G} \alpha$ или $S_2 \overset{*}{\lhd_G} \alpha$. Но понеже не добавяме други правила с V_1 и V_2, тогава $S_1 \overset{*}{\lhd_{G_1}} \alpha$ или $S_2 \overset{*}{\lhd_{G_2}} \alpha$. Така $\alpha = \alpha_1\alpha_2 \in \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$. Обратно, ако $\alpha \in \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$ то $S_1 \overset{*}{\lhd_{G_1}} \alpha$ или $S_2 \overset{*}{\lhd_{G_2}} \alpha$, откъдето $S_1 \overset{*}{\lhd_G} \alpha$ или $S_2 \overset{*}{\lhd_G} \alpha$, и понеже $S \to_G S_1 \mid S_2$, $S \overset{*}{\lhd} \alpha$. Получаваме, че $\mathcal{L}(G) = \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$.
- $S \to_G S_1S_2$ заедно с правилата R_1, R_2 ще генерира езикът $\mathcal{L}(G_1) \cdot \mathcal{L}(G_2)$.
  Ако $S \overset{*}{\lhd_G} \alpha$ и $\alpha \in \Sigma^*$, то тогава очевидно понеже $S \to_G S_1S_2$ е единственото правило, $S_i \overset{*}{\lhd_G} \alpha_i$ $(i = 1, 2)$ и $\alpha = \alpha_1\alpha_2$. Но понеже не добавяме други правила с V_1 и V_2, $S_i \overset{*}{\lhd_{G_i}} \alpha_i$ за $i = 1, 2$. Така $\alpha = \alpha_1\alpha_2 \in \mathcal{L}(G_1) \cdot \mathcal{L}(G_2)$. Обратно, ако $\alpha_i \in \mathcal{L}(G_i)$ за $i = 1, 2$, то $S_i \overset{*}{\lhd_{G_i}} \alpha_i$, откъдето понеже $S \to_G S_1S_2$, $S \overset{*}{\lhd} \alpha_1\alpha_2$. Получаваме, че $\mathcal{L}(G) = \mathcal{L}(G_1) \cdot \mathcal{L}(G_2)$.
- $S \to_G SS_i \mid \varepsilon$ заедно с правилата R_i ще генерира езикът $\mathcal{L}(G_i)^*$ за $i = 1, 2$.
  Тук идеята е подобна на тази при конкатенацията. За интуиция доста помага факта, че $L^* = (L^* \cdot L) \cup \{\varepsilon\}$. Пълното доказателство ще оставим за упражнение на читателя.

Вече имайки, че регулярните операции запазват безконтекстност, индукцията е завършена. □

Вече можем да започнем да си мислим как изглежда множеството от различните видове езици, които познаваме:



По нататък ще покажем, че тази картинка не е подвеждаща т.е. има небезконтекстни езици.

3.2 Регулярни граматики и еквивалентност със автоматите

Тук ще направим нещо, което на пръв поглед може да изглежда безсмислено. Ще покажем частен случай на безконтекстните граматики, които са еквивалентни на автоматите като изразителна мощ.

Дефиниция 3.2.1. Една безконтекстна граматика $G = \langle \Sigma, V, S, R \rangle$ ще наричаме **регулярна**, ако единствените правила в G са от вида:

- $A \to aB$ за някои $A, B \in V$, $a \in \Sigma$
- $A \to \varepsilon$ за някое $A \in V$

Това изглежда много подобно на дефиницията на автомат. Тук обаче вместо от състояние със буква да отидем в друго състояние, просто заместваме старото състояние с буквата и новото състояние. В единия случай четем символи, а в другия ги генерираме. Генерирането на ε е аналога на това да спрем да четем думата (почти).

Твърдение 3.2.2. *За всеки ДКА $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ има регулярна граматика $G = \langle \Sigma, V, S, R \rangle$ с $\mathcal{L}(G) = \mathcal{L}(\mathcal{A})$*

Доказателство. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е ДКА. Нека $Q = \{q_1, \ldots, q_n\}$ като $q_1 = s$.
Строим граматика $G = \langle \Sigma, V, S, R \rangle$ за $\mathcal{L}(\mathcal{A})$:

- $V = \{P_1, \ldots, P_n\}$ (на всяко състояние q_i съответства променлива P_i)
- $S = P_1$ ($q_1 = s$)
- Ако $\delta(q_i, x) = q_j$, то $P_i \to xP_j$
- Ако $q_i \in F$, то $P_i \to \varepsilon$

Твърдението, което трябва да покажем е, че за всяко $\alpha \in \Sigma^*$, $\delta^*(q_i, \alpha) \in F \iff P_i \overset{*}{\lhd} \alpha$. В посоката ($\Rightarrow$) се прави индукция по дължината на $|\alpha|$, а в посоката ($\Leftarrow$) се прави индукция по дефиницията на $\overset{n}{\lhd}$. Това остава за читателя. Остават само довършителни разсъждения:

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}) &\iff (\exists i \in \{1, \ldots, n\})\, (\delta^*(q_1, \alpha) = q_i \,\&\, q_i \in F) \\
&\iff (\exists i \in \{1, \ldots, n\})\, (P_1 \overset{*}{\lhd} \alpha P_i \,\&\, P_i \to \varepsilon) \\
&\iff (\exists i \in \{1, \ldots, n\})\, (P_1 \overset{*}{\lhd} \alpha P_i \,\&\, P_i \overset{1}{\lhd} \varepsilon) \\
&\overset{!!!}{\iff} P_1 \overset{*}{\lhd} \alpha \\
&\iff \alpha \in \mathcal{L}(G)
\end{aligned}$$

Внимание (!!!)**.** По принцип разсъждения, подобни на ($\Leftarrow$) не са верни в общият случай. Тук можем да си го позволим сега поради природата на нашата граматика. Единственото нещо, което може да правят променливите P_i е да генерират думи от вида αP_j. Разсъждения, подобни на ($\Rightarrow$) обаче, ще бъдат верни, поради Твърдение 3.1.8.

□

Твърдение 3.2.3. *За всяка регулярна граматика $G = \langle \Sigma, V, S, R \rangle$ има НКА $\mathcal{N} = \langle \Sigma, Q, S, \Delta, F \rangle$ с $\mathcal{L}(\mathcal{N}) = \mathcal{L}(G)$*

Доказателство. Нека $G = \langle \Sigma, V, S, R \rangle$ е регулярна граматика. Нека $V = \{A_1, \ldots, A_n\}$, като $A_1 = S$.
Строим НКА $\mathcal{N} = \langle \Sigma, Q, I, \Delta, F \rangle$ за $\mathcal{L}(G)$:

- $Q = \{q_1, \ldots, q_n\}$
- $I = \{q_1\}$
- $\Delta(q_i, x) = \{q_j \in Q \mid A_i \to xA_j\}$
- $F = \{q_i \in Q \mid A_i \to \varepsilon\}$

Ще формулираме само помощното твърдение и ще оставим всичко останало на читателя:

$$\Delta^*(q_i, \alpha) \cap F \neq \varnothing \iff A_i \overset{*}{\lhd} \alpha \,\&\, \alpha \in \Sigma^*$$

□

С тези две твърдения не направихме много. Показахме, че има някакъв вид безконтекстни граматики със изразителната мощ на автоматите. Това по принцип не е лошо нещо, но самата граматика е доста подобна на автомата, така че не е и като да имаме напълно нов начин за изразяване на регулярни езици. По ценното тук е да се види самата "линейност" на извода на регулярната граматика. Можем тази техника да я разширим и да я адаптираме за по-сложни задачи.
Нека да си помислим, какво ще стане ако приложим конструкцията от Твърдение 3.2.2, но вместо $P_i \to xP_j$ имаме $P_j \to P_jx$. Ще генерираме думите на $\mathcal{L}(\mathcal{A})$, но в обратен ред т.е. $\mathcal{L}(G) = \mathcal{L}(\mathcal{A})^{rev}$.

Задача 3.2.4. *Да се опише подробно конструкцията за* $\mathcal{L}(\mathcal{A})^{rev}$.

Какво пък ще стане, ако решим да сложим $P_i \to xP_jx$ вместо $P_i \to xP_j$. Ще генерираме от ляво дума от $\mathcal{L}(\mathcal{A})$, а от дясно същата дума но в обратен ред. Тогава $\mathcal{L}(G) = \{\alpha\alpha^{rev} \mid \alpha \in \mathcal{L}(\mathcal{A})\}$

Задача 3.2.5. *Да се опише подробно конструкцията за* $\{\alpha\alpha^{rev} \mid \alpha \in \mathcal{L}(\mathcal{A})\}$.

Можем и да станем по креативни:

Твърдение 3.2.6. *Нека* L *е регулярен. Тогава езикът* $\text{Move}(L) = \{a^nb^m \mid (\exists\alpha \in L)\,(|\alpha|_a = n \,\&\, |\alpha|_b = m)\}$ *е безконтекстен.*

Доказателство. Ще покажем само конструкцията и ще формулираме помощно твърдение; другото е за читателя.
Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F\rangle$ е ДКА. Нека $Q = \{q_1, \ldots, q_n\}$ като $q_1 = s$.
Строим граматика $G = \langle \Sigma, V, S, R\rangle$ за $\text{Move}(L)$:

- $V = \{P_1, \ldots, P_n\}$ (на всяко състояние q_i съответства променлива P_i)
- $S = P_1$ ($q_1 = s$)
- Ако $\delta(q_i, a) = q_j$, то $P_i \to aP_j$ (слагаме буквите a от ляво)
- Ако $\delta(q_i, b) = q_j$, то $P_i \to P_jb$ (слагаме буквите b от дясно)
- Ако $q_i \in F$, то $P_i \to \varepsilon$

Трябва да се покаже, че:

- $\delta^*(q_i, \alpha) \in F \Rightarrow P_i \overset{*}{\lhd} a^{|\alpha|_a}b^{|\alpha|_b}$
- $P_i \overset{*}{\lhd} \alpha \,\&\, \alpha \in \Sigma^* \Rightarrow (\exists\beta \in \Sigma^*)(\delta^*(s, \beta) \in F \,\&\, \alpha = a^{|\beta|_a}b^{|\beta|_b})$

□

Задача 3.2.7. *Нека* L *е регулярен. Да се покаже, че е безконтекстен езикът:*

$$\text{AltMove}(L) = \{a^nb^m \mid (\exists\alpha \in L)\,(|\alpha|_a = m \,\&\, |\alpha|_b = n)\}$$

Упътване: да се направи дребна модификация на конструкцията от Твърдение 3.2.6

Задача 3.2.8. *Нека* L *е регулярен и нека* L_1, L_2 *са безконтекстни. Да се докаже, че са безконтекстни следните езици:*

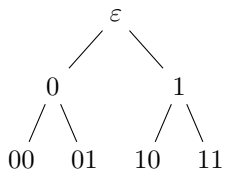
- $L' = \{\alpha_1 \ldots \alpha_n\beta_1 \ldots \beta_m \mid (\exists\alpha \in L)\,(|\alpha|_a = n \,\&\, |\alpha|_b = m) \,\&\, \alpha_i \in L_1 \,\&\, \beta_i \in L_2\}$
- $L'' = \{\alpha_1 \ldots \alpha_{2n}\beta_1 \ldots \beta_{3m} \mid (\exists\alpha \in L)\,(|\alpha|_a = n \,\&\, |\alpha|_b = m) \,\&\, \alpha_i \in L_1 \,\&\, \beta_i \in L_2\}$
- $L''' = \{\beta_1 \ldots \beta_m\alpha_1 \ldots \alpha_n \mid (\exists\alpha \in L)\,(|\alpha|_a = n \,\&\, |\alpha|_b = m) \,\&\, \alpha_i \in L_1 \,\&\, \beta_i \in L_2\}$

Упътване: да се адаптират техники от предишни задачи

3.3 Дървета на извод

Дефиниция 3.3.1. Нека Σ е крайна азбука и нека $T \subseteq \Sigma^*$ е краен език. T ще наричаме **дърво над** Σ, ако $\text{Pref}(T) = T$.

Това на пръв поглед изглежда доста странна дефиниция на дърво. Ще се опитаме да си дадем интуиция за нея. Можем да си мислим за всяка дума като за връх, а ребрата са между върхове от вида α и αx, където $\alpha \in \Sigma^*$, $x \in \Sigma$:



Тук $T = \Sigma^{\leq 2}$. Всяка дума $\alpha \in T$ е връх на T. Има ребро между 1 и 10 защото са на "еднобуквена конкатенация разстояние". Понеже не може да се каже същото за двойки върхове като $\langle \varepsilon, 11 \rangle$ или $\langle 11, 00 \rangle$, между тях ребра няма.
Ще бъде много удобно да въведем някои познати термини за дървета:

Дефиниция 3.3.2. Нека $T \subseteq \Sigma^*$ е дърво. Дефинираме следните понятия:

- **височината на** T ще наричаме $\text{height}(T) = \max\{|\alpha| \mid \alpha \in T\}$
- **децата на връх** $\alpha \in T$ ще наричаме $\text{children}(\alpha) = (\{\alpha\} \cdot \Sigma) \cap T$
- **листата на** T ще бележим с $\text{leaves}(T) = \{\alpha \in T \mid \text{children}(\alpha) = \varnothing\}$

Забележка. За следващата дефиниция ще приемаме, че зад всяка крайна азбука $\Sigma = \{x_1, \ldots, x_n\}$ стои линейна наредба, за да можем да говорим за лексикографска наредба на думите. Нея ще бележим с $<$.

Дефиниция 3.3.3. Нека $G = \langle \Sigma, V, S, R \rangle$ е безконтекстна граматика. Нека $T \subseteq \Sigma^*$ е дърво и нека $\lambda : T \to (\Sigma_\varepsilon \cup V)$. Ще наричаме $\mathcal{T} = \langle T, \lambda \rangle$ **дърво на извод, съгласувано с** G, ако:

- $\lambda(\varepsilon) \in \Sigma \cup V$
- ако $\text{children}(\alpha) = \{\alpha x_1, \ldots, \alpha x_k\}$ като $x_1 < \cdots < x_k$, то трябва да имаме правилото $\lambda(\alpha) \to_G \lambda(\alpha x_1) \ldots \lambda(\alpha x_k)$ като $\lambda(\alpha x_i) \neq \varepsilon$
- ако $\text{children}(\alpha) = \{\alpha x\}$ и имаме правилото $\lambda(\alpha) \to_G \varepsilon$, то може $\lambda(\alpha x) = \varepsilon$

Освен това:

- $\text{root}(\mathcal{T}) = \lambda(\varepsilon)$ - това е **коренът на** $\mathcal{T}$
- $\text{word}(\mathcal{T}) = \alpha_1 \ldots \alpha_n$, където $\text{leaves}(T) = \{\alpha_1, \ldots, \alpha_n\}$ и $\alpha_1 < \cdots < \alpha_n$

Можем да си мислим, че множеството T задава "формата" на дървото, а функцията λ слага като "етикет" променлива, буква или ε, като всичко това е съгласувано с правилата в G.

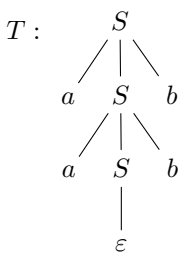
Забележка. Ще изневерим на нашата нотация. Вместо да казваме:

"Нека $\mathcal{T} = \langle T, \lambda \rangle$ е дърво на извод..."

за по-кратко ще казваме:

"Нека T е дърво на извод..."

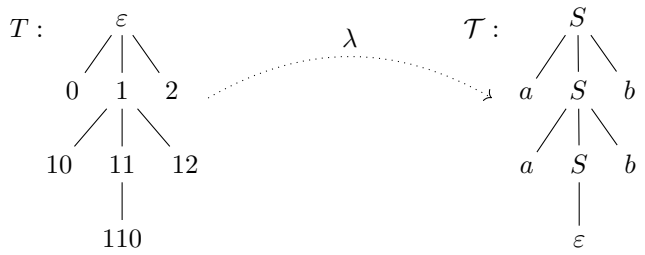
Под $\text{word}(T)$ ще имаме предвид $\text{word}(\mathcal{T})$, и под $\text{root}(T)$ ще имаме предвид $\text{root}(\mathcal{T})$.
Нека вземем граматиката с правилата $S \to aSb \mid \varepsilon$. Примерно дърво на извод би било:



За дървото на извод T имаме, че:

- $\text{word}(T) = aabb$
- $\text{height}(T) = 3$
- $\text{root}(T) = S$.

Внимание. Ако караме по дефиницията това, което наистина се случва отзад, е малко по-различно:



Както споменахме преди малко, T всъщност е само формата, а вече функцията λ дава семантиката на дървото.

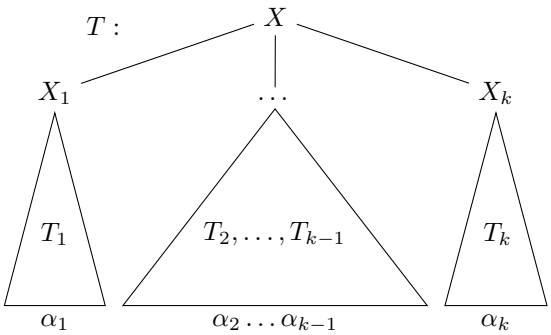
Твърдение 3.3.4. *Нека $G = \langle \Sigma, V, S, R \rangle$ е безконтекстна граматика и нека $X \in \Sigma \cup V$. Тогава:*

$$X \overset{l}{\lhd} \alpha \iff \textit{има дърво на извод } T \textit{ такова, че } \text{root}(T) = X,\ \text{height}(T) = l \textit{ и } \text{word}(T) = \alpha$$

Доказателство. И двете твърдения доказваме с индукция.

($\Rightarrow$) С индукция по l:

- $X \overset{0}{\lhd} X$. Очевидно има дърво на извод с единствен връх X ✓
- $X \overset{l+1}{\lhd} \alpha$. Тогава $X \to_G X_1 \dots X_k$, $X_1 \overset{l_1}{\lhd} \alpha_1, \dots, X_k \overset{l_k}{\lhd} \alpha_k$ като $\alpha = \alpha_1 \dots \alpha_k$ и $l = \max\{l_1, \dots, l_k\}$. По ИП има дървета на извод $T_1, \dots, T_k$ такива, че $\text{root}(T_i) = X_i$, $\text{height}(T_i) = l_i$ и $\text{word}(T_i) = \alpha_i$. Можем да построим дърво на извод T с корен T, като неговите преки наследници са корените на дърветата $T_1, \dots, T_k$:



  За знаем, че $\text{root}(T) = X$, $\text{height}(T) = 1 + \max\{\text{height}(T_1), \dots, \text{height}(T_k)\} = 1 + \max\{l_1, \dots, l_k\} = l + 1$ и $\text{word}(T) = \alpha_1 \dots \alpha_k = \alpha$.

($\Leftarrow$) С индукция по $\text{height}(T)$:

- Единственото дърво T с $\text{root}(T) = X$ и $\text{height}(T) = 0$ е дървото само с връх X, но $X \overset{0}{\lhd} X = \text{word}(T)$ ✓
- Ако T е дърво с корен X и единствен наследник ε, то тогава $\text{height}(T) = 1$ и има правилото $X \to_G \varepsilon$. Понеже $X \overset{0}{\lhd} X$, $X \overset{1}{\lhd} \varepsilon = \text{word}(T)$. Ако пък T е дърво с корен X и преки наследници $X_1, \dots, X_k$, като ще наричаме поддърветата вкоренени в тях $T_1, \dots, T_k$. Нека $\text{word}(T) = \alpha$, $\text{word}(T_i) = \alpha_i$, $\text{height}(T_i) = l_i$. Ясно е, че $\alpha = \alpha_1 \dots \alpha_k$ и $\text{height}(T) = 1 + \max\{l_1, \dots l_k\}$. По ИП $X_i \overset{l_i}{\lhd} \alpha_i$. Тъй като $X_1, \dots, X_k$ са преки наследници на корена X на T, то тогава има правило $X \to X_1 \dots X_k$, откъдето $X \overset{l+1}{\lhd} \alpha_1 \dots \alpha_k = \alpha$.

□

С това доказахме, че $\mathcal{L}(G) = \{\alpha \in \Sigma^* \mid$ има дърво на извод T съглавувано с G и $\text{root}(T) = S,\ \text{word}(T) = \alpha\}$ за всяка безконтекстна граматика $G = \langle \Sigma, V, S, R\rangle$. Вече вместо за $\overset{*}{\lhd}$ да си мислим просто като за една релация, можем да си мислим за дървета на извод. Зад нея вече стои някаква "по-смислена" семантика.

Задача 3.3.5. *Нека граматиката $G = \langle \Sigma, \{S, A, B, C\}, S, R\rangle$ се определя чрез следните правила:*

$$S \to A \mid BA \mid BCAC \mid CC$$
$$A \to AaB \mid bbC \mid aCa \mid a$$
$$B \to baA \mid ACA \mid \varepsilon$$
$$C \to AbCb \mid a \mid b$$

Да се построят дърветата на извод за думите $\alpha_1 = aaabbb, \alpha_2 = baaabbbaa, \alpha_3 = aabbbaabbbaabbb$.
Упътване: за α_1 има дърво на извод с корен C, за α_2 с корен B и за α_3 с корен S.

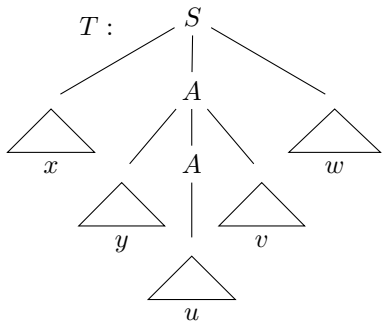
3.4 Небезконтекстни езици

Сега ще покажем съществуването на небезконтекстни езици. За целта ще разгледаме безконтекстната версия на Лема за покачването:

Лема 3.4.1 (Лема за покачването)**.** *Нека L е безконтекстен език. Тогава:*

$$(\exists p \geq 1)$$
$$(\forall \alpha \in L,\ |\alpha| \geq p)$$
$$(\exists x, y, u, v, w \in \Sigma^*,\ xyuvw = \alpha,\ |yuv| \leq p,\ |yv| \geq 1)$$
$$(\forall i \in \mathbb{N})[xy^iuv^iw \in L]$$

Доказателство. Нека L е безконтекстен език. Нека $G = \langle \Sigma, V, S, R\rangle$ е безконтекстна граматика за L.
Полагаме $b = \max\{|\gamma| \mid A \to_G \gamma$ е правило в $G\}$ и $p = b^{|V|+1}$. За b можем да си мислим, че това е най-голямата разклоненост на дърво на извод. Няма как да има връх в дърво на извод с повече от b брой преки наследници. Тогава едно дърво на извод в G с височина h не може да има дума с дължина повече от b^h. Така ако $\alpha \in L$ и $|\alpha| \geq p$, в което и да е дърво на извод T такова, че $\text{word}(T) = \alpha$, имаме $\text{height}(T) \geq |V|$. Тогава в пътя на едно такова дърво T от корена до някои листа ще има повторение на променливи. Нека T е дърво на извод за α с минимален брой елементи. Знаем, че има повторение на някаква променлива $A \in V$. Нека това повторение е възможно най-надолу. Със сигурност това ще бъде измежду последните $|V|$ върха в някои път между корена и листата. На картинка имаме следното:



Понеже срещанията на a се случват достатъчно надолу, $|yuv| \leq p$. Понеже дървото T е с минимален брой върхове $|yv| \geq 1$. Ако това не беше вярно, $yv = \varepsilon$ и можехме да премахнем първото срещане на A, с което строим по-малко дърво на извод. Остава само да проверим, че $xy^iuv^iw \in L$ за всяко $i \in \mathbb{N}$ $(\star)$. От Твърдение 3.3.4, $A \overset{*}{\lhd} yAv$. Ще покажем с индукция по i че $A \overset{*}{\lhd} y^iAv^i$ за всяко $i \in \mathbb{N}$.

- $A \overset{0}{\lhd} A = y^0Av^0$ ✓
- Нека $A \overset{*}{\lhd} y^iAv^i$, тогава понеже $A \overset{*}{\lhd} yAv$, $A \overset{*}{\lhd} y^iyAvv^i = y^{i+1}Av^{i+1}$ (от Твърдение 3.1.8)

От Твърдение 3.3.4, $A \overset{*}{\lhd} u$, откъдето използвайки $(\star)$ получаваме, че $A \overset{*}{\lhd} y^iuv^i$ за всяко $i \in \mathbb{N}$. Също така $S \overset{*}{\lhd} xAw$, откъдето $S \overset{*}{\lhd} xy^iuv^i$ за всяко $i \in \mathbb{N}$, с което сме готови. □

Разбира се, ние ще използваме по често следствието от лемата.

Следствие 3.4.2 (Лема за покачването (контрапозиция))**.** *Ако е изпълнено, че:*

$$(\forall p \geq 1)$$
$$(\exists \alpha \in L,\ |\alpha| \geq p)$$
$$(\forall x, y, u, v, w \in \Sigma^*,\ xyuvw = \alpha,\ |yuv| \leq p,\ |yv| \geq 1)$$
$$(\exists i \in \mathbb{N})[xy^iuv^iw \notin L]$$

то тогава L не е безконтекстен.

Започваме с каноничния пример за небезконтекстен език:

Твърдение 3.4.3. *Езикът $L = \{a^nb^nc^n \mid n \in \mathbb{N}\}$ не е безконтекстен.*

Доказателство. Нека $p \geq 1$. Нека $\alpha = a^pb^pc^p$. Ясно е, че $\alpha \in L$ и $|\alpha| \geq p$. Нека $x, y, u, v, w \in \Sigma^*$ са такива, че $xyuvw = \alpha$, $|yuv| \leq p$, $|yv| \geq 1$. Очевидно няма как $yv = a^nc^m$ за $n, m > 0$. Тогава имаме няколко варианта за yv:

1 сл. $yv = a^t$ за някое $1 \leq t \leq p$. Тогава $xy^0uv^0w = xuw = a^{p-t}b^pc^p \notin L$ $(p - t < p)$

2 сл. $yv = b^t$ за някое $1 \leq t \leq p$ - аналогично на 1 сл.

3 сл. $yv = c^t$ за някое $1 \leq t \leq p$ - аналогично на 1 сл.

4 сл. $yv = a^{t_1}b^{t_2}$ за някое $1 \leq t_1, t_2 \leq p$, $t_1 + t_2 \leq p$. Тогава $xy^0uv^0w = xuw = a^{p-t}b^{p-t}c^p \notin L$ $(p - t < p)$

5 сл. $yv = b^{t_1}c^{t_2}$ за някое $1 \leq t_1, t_2 \leq p$, $t_1 + t_2 \leq p$ - аналогично на 4 сл.

От Лема за покачването (контрапозиция) следва, че L не е безконтекстен. □

Задача 3.4.4. *Да се докаже, че следните езици не са безконтекстна:*

- $L_1 = \{a^nb^{2n}c^n \mid n \in \mathbb{N}\}$
- $L_2 = \{a^nb^{3n}c^{5n} \mid n \in \mathbb{N}\}$
- $L_3 = \{a^nb^mc^n \mid n \leq m\}$
- $L_4 = \{a^nb^mc^k \mid n \leq m \leq k\}$

Упътване: за L_1 и L_2 да се адаптира решението от Твърдение 3.4.3, а за L_3 и L_4 трябва да се използват думи от същия вид.

Внимание. За тази версия на лемата, е много по-важен избора на дума. Тук трябва да се стремим думата да е възможно най-близко до това да излезне от езика. Да вземем за пример L_3 от Задача 3.4.4. Ако вземем дума от вида $a^pb^{p+t}c^p$, за $p, t \geq 1$, то ако $yv = b$, думата няма да излезе от L_3 каквото и да правим. Подобни разсъждения могат да се направят и за L_4. Когато става въпрос за небезконтекстност, трябва да сме по-нежни и по-деликатни.

Задача 3.4.5. *Да се докаже, че следните езици не са безконтекстни:*

- $L_1 = \{\alpha\alpha \mid \alpha \in \Sigma^*\}$
- $L_2 = \{\alpha\alpha\alpha \mid \alpha \in \Sigma^*\}$
- $L_3 = \{\alpha\alpha^{rev}\alpha \mid \alpha \in \Sigma^*\}$

Твърдение 3.4.6. *Операцията сечение и допълнение не запазват безкотекстност*

Доказателство. За сечението елементарен контрапример:

$$\underbrace{\{a^nb^nc^k \mid n, k \in \mathbb{N}\}}_{L_1-\text{безконтекстен}} \cap \underbrace{\{a^nb^kc^k \mid n, k \in \mathbb{N}\}}_{L_2-\text{безконтекстен}} = \underbrace{\{a^nb^nc^n \mid n \in \mathbb{N}\}}_{L-\text{небезконтекстен}}$$

Ако допуснем, че безконтекстните езици са затворени относно допълнение, то тогава понеже L_1 и L_2 са безконтекстни, $\overline{L_1} \cup \overline{L_2}$ също е безконтекстен (обединението запазва безконтекстност). Можем да приложим допълнение още веднъж и ще получим отново безконтекстен език, откъдето $\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2 = L$ е безконтекстен, което е противоречие. □

Тук се вижда как въпреки че разпознаваме видове езици, губим някои свойства. Освен, че губим операции като допълнение и сечение, ние също така губим константната памет, която имахме при автоматите.

Внимание. Въпреки, че в общия случай не можем да приемем, че допълнението на безконтекстен език е също е безконтекстен език, понякога това е вярно. Очевидно е вярно за регулярните езици, но не само и аз тях. Има безконтекстни езици, които не са регулярни, чието допълнение е безконтекстен език:

Твърдение 3.4.7. *Езикът $L = \Sigma^* \setminus \{a^n b^n \mid n \in \mathbb{N}\}$ е безконтекстен.*

Доказателство. Само ще покажем представяне на езика, което очевидно е безконтекстно. Подробностите оставяме на читателя. Нека помислим какво би означавало $\alpha \in L$. Има два варианта:

1 сл. $\alpha = a^n b^k$ и $n \neq k$. Това би означавало, че $\alpha \in \underbrace{\{a^n b^k \mid n > k\}}_{L_1} \cup \underbrace{\{a^n b^k \mid n < k\}}_{L_2}$.

2 сл. α изобщо не е от $\{a\}^*\{b\}^*$. За да не е вярно това някъде в думата има буква b преди буква a, откъдето $\alpha \in \underbrace{\Sigma^* \cdot \{b\} \cdot \Sigma^* \cdot \{a\} \cdot \Sigma^*}_{L_3}$.

Така можем да представим L като обединение на безконтекстни езици:

$$L = \underbrace{(\{a\}^+ \cdot \{a^n b^n \mid n \in \mathbb{N}\})}_{L_1-\text{безконтекстен}} \cup \underbrace{(\{a^n b^n \mid n \in \mathbb{N}\} \cdot \{b\}^+)}_{L_2-\text{безконтекстен}} \cup \underbrace{(\Sigma^* \cdot \{b\} \cdot \Sigma^* \cdot \{a\} \cdot \Sigma^*)}_{L_3-\text{безконтекстен}}$$

□

3.5 Нормална форма на Чомски

Тук ще покажем, че всеки безконтекстен език с точност до липсата на ε може да се представи чрез граматика във "хубав" вид. За целта постъпково ще си "опростяваме" нашата граматика т.е. ще гледаме да променим свойствата на граматика запазвайки езика (с точност до липсата на ε).

Премахване на дългите правила

Под дълго правило имаме предвид $X \to X_1 \dots X_k$, където $X_i \in \Sigma \cup V$ и $k \geq 3$. За да премахнем едно такова правило просто добавяваме променливите $Y_1, \dots, Y_{k-2}$ и заменяме горното правило с:

$$X \to X_1 Y_1,\ Y_1 \to X_2 Y_2,\ \dots,\ Y_{k-2} \to X_{k-1} X_k$$

Прилагайки тази конструкция за всички правила получаваме граматика със същия език, която има само правила от вида $A \to \beta$, където $|\beta| \leq 2$.

Премахване не ε правилата

Целта ни ще бъде да премахнем всички правила от вида $X \to \varepsilon$ като запазим езика (без ε). Ще направим рекурсия:

- $\text{Eps}(0) = \varnothing$
- $\text{Eps}(n+1) = \{A \in V \mid (\exists \beta \in \text{Eps}(n)^*)\ (A \to \beta)\}$

Първо в $\text{Eps}(1)$ ще бъдат променливите, които генерират ε, после в $\text{Eps}(2)$ ще бъдат променливите, които генерират тях и т.н. Лесно може да се покаже, че:

$$\text{Eps}(n) = \{A \in V \mid A \overset{\leq n}{\lhd} \varepsilon\}$$

Тъй като $\text{Eps}(n) \subseteq \text{Eps}(n+1) \subseteq V$, то от някъде нататък ще получаваме едно и също множество. Нека това множество бележим с Eps. Нека сега променим правилата. Ако има правило $X \to X_1 \dots X_k$, при условието че $X_1 \dots X_k \neq \varepsilon$, добавяме всички правила от вида $X \to Y_1 \dots Y_k$, където:

- ако $X_i \notin \text{Eps}$, то $Y_i = X_i$
- ако $X_i \in \text{Eps}$, то $Y_i = X_i$ или $Y_i = \varepsilon$

Очевидно е, че в новата граматика няма да има ε правила.

Премахване на преименуващите правила

Целта ни тук ще бъде да премахнем правила от вида $A \to B$. Обаче тогава трябва да видим какви думи генерира B и по някакъв начин да ги добавим към тези, които генерира A. Отново правим рекурсия:

- $\text{Rename}(0) = V \times V$
- $\text{Rename}(n+1) = \text{Rename}(n) \cup \{\langle A, C\rangle \mid (\exists B \in V)\,(A \to B \,\&\, \langle B, C\rangle \in \text{Rename}(n))\}$

Цялата идея е да се види колко далече може да стигне една променлива с преименуване. Лесно може да се съобрази, че:

$$\text{Rename}(n) = \{\langle A, B\rangle \mid A \overset{\leq n}{\lhd} B\}$$

Тъй като $\text{Rename}(n) \subseteq \text{Rename}(n+1) \subseteq V \times V$, то от някъде нататък ще получаваме едно и също множество. Нека това множество бележим с Rename. Вече можем да кажем кои ще са новите правила в граматиката:

$$R_{norename} = \{\langle A, \beta\rangle \in V \times (\Sigma \cup V)^* \mid (\exists B \in V)\,(\underbrace{\langle A, B\rangle \in \text{Rename}}_{A \text{ може да се замени с } B} \;\&\; \underbrace{\langle B, \alpha\rangle \in R \setminus (V \times V)}_{B \to \alpha \text{ не е преименуващо правило}})\}$$

Премахване на правилата, които генерират повече от 1 буква

Ако имаме правило от вида $X \to x_1x_2$, където $x_1, x_2 \in \Sigma \cup V$. За всяко $x_i \in \Sigma$ можем да добавим нова променлива X_i с правилото $X_i \to x_i$ и да заменим в предното правило x_i със X_i. Например за правилото $A \to bc$ можем да добавим нови променливи B и C, заменяйки старото правило с правилата:

$$A \to BC,\; B \to b,\; C \to c$$

Дефиниция 3.5.1. Една безконтекстна граматика $G = \langle \Sigma, V, S, R\rangle$ е в **нормална форма на Чомски** (накратко НФЧ), ако всичките и правила са от вида:

- $A \to BC$ за някои $A, B, C \in V$
- $A \to a$ за някои $A \in V$, $a \in \Sigma$

Прилагайки тези алгоритми в тази последователност, получаваме при вход безконтекстна граматика G получаваме граматика G' в НФЧ с $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

Забележка. Ако искаме да добавим ε в езика можем да го направим много лесно. Добавяме нова променлива S_0 и правилата $S_0 \to \varepsilon$ заедно с $S_0 \to \alpha$ за всяко правило $S \to \alpha$. Тогава правилата ще бъдат от вида:

- $S \to \varepsilon$
- $A \to BC$ за някои $A, B, C \in V$ като $B, C \neq S$
- $A \to a$ за някои $A \in V$, $a \in \Sigma$

За простота няма да се занимаваме с този вид граматика. Ще приемем, че случаите, в които участва ε са тривиални са разглеждане, и няма да се занимаваме с тях.

Твърдение 3.5.2. *За всеки безконтекстен език L, езикът $sub(L)$ (от Дефиниция 2.10.2) също е безконтекстен.*

Доказателство. Само ще покажем конструкцията и откъде идва удобството от НФЧ при конструкции, като пълното доказателство остава за читателя.
Ще е хубаво да имаме предвид някои хубави свойства на поддумите:

- ако α е поддума на β можем да си мислим как "задраскваме" някои букви от β и получаваме α
- $sub(L_1 \cdot L_2) = sub(L_1) \cdot sub(L_2)$

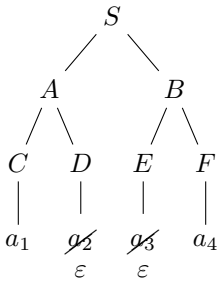
А имайки граматика в НФЧ можем да разсъждаваме така:

- ако си мислим изолирано за правилата от вида $A \to a$ можем много лесно да направим "задраскването"
- за правилата от вида $A \to BC$ можем да си мислим, че към $\mathcal{L}_G(A)$ се добавя $\mathcal{L}_G(B) \cdot \mathcal{L}_G(C)$

Сега нека опишем вече конструкцията. Нека $G = \langle \Sigma, V, S, R \rangle$ е граматика в НФЧ за L. В новата граматика G_{sub}, която строим ще имаме същите променливи, същата начална променлива, като правилата са следните:

- имаме същите правила от G
- ако $A \to a$ е правило в G, то добавяме правилото $A \to \varepsilon$ (тук става "задраскването")

Сега да опишем на картинка какво ще направим. Да кажем, че генерираме думата $a_1a_2a_3a_4$ в оригиналния език. Искаме в новия език да можем да генерираме думата a_1a_4:



Понеже сме имали правилото $D \to a_2$ сме добавили правилото $D \to \varepsilon$, същото за E и a_3. Това ни позволи да премахнем тези букви.
Ако искаме да сме подробни, можем да покажем, че за всички $A \in V$, $\alpha \in \Sigma$ и $n \in \mathbb{N}$:

- ако $A \overset{n}{\lhd}_G \alpha$, то $A \overset{n}{\lhd}_{G_{sub}} \beta$ за всяка поддума β на α
- ако $A \overset{n}{\lhd}_{G_{sub}} \alpha$, то има β такава, че $A \overset{n}{\lhd}_G \beta$ и α е поддума на β.

□

Твърдение 3.5.3. *За всеки безконтекстен език L езикът L^{rev} е безконтекстен.*

Доказателство. Нека $G = \langle \Sigma, V, S, R \rangle$ е граматика в НФЧ за L. В новата граматика G_{rev} за L^{rev} имаме същите променливи и начална променлива. Новите правила са следните:

- ако $A \to_G a$, то тогава $A \to_{G_{rev}} a$
- ако $A \to_G BC$, то тогава $A \to_{G_{rev}} CB$

Използвайки, че $(\beta_1\beta_2)^{rev} = \beta_2^{rev}\beta_1^{rev}$, лесно се показва, че за всички $A \in V$, $\alpha \in \Sigma$ и $n \in \mathbb{N}$

$$A \overset{n}{\lhd}_G \alpha \iff A \overset{n}{\lhd}_{G_{rev}} \alpha^{rev}$$

И двете посоки са аналогични, за това ще направим само едната:

- ако $A \overset{0}{\lhd}_G \alpha$, то $\alpha = A \notin \Sigma^*$ ✓
- ако $A \overset{n+1}{\lhd}_G \alpha$, то сме приложили правило:

  1 сл. приложили сме правило от вида $A \to_G a$:
  Тогава $\alpha = a$, $n = 0$ и $A \to_{G_{rev}} a$, откъдето $A \overset{1}{\lhd}_{G_{rev}} a = \alpha$

  2 сл. приложили сме правило от вида $A \to_G BC$:
  Тогава $B \overset{n_1}{\lhd}_G \alpha_1$, $C \overset{n_2}{\lhd}_G \alpha_2$, като $\alpha = \alpha_1\alpha_2$ и $n = \max\{n_1, n_2\}$. Тогава по ИП, $B \overset{n_1}{\lhd}_{G_{rev}} \alpha_1^{rev}$ и $C \overset{n_2}{\lhd}_{G_{rev}} \alpha_2^{rev}$.
  Също така има правилото $A \to_{G_{rev}} CB$, откъдето $A \overset{n+1}{\lhd} \alpha_2^{rev}\alpha_1^{rev} = (\alpha_1\alpha_2)^{rev}$.

Показвайки твърдението получаваме, че $\mathcal{L}_{G_{rev}}(A) = \mathcal{L}_G(A)^{rev}$ за всяка променлива A, в частност:

$$\mathcal{L}(G_{rev}) = \mathcal{L}_{G_{rev}}(S) = \mathcal{L}_G(S)^{rev} = \mathcal{L}(G) = L^{rev}$$

□

Хубавото на такива конструкции е, че промените, които правим не са големи. Тук дърветата на извод имат същата "форма". Разбира се, при сложни примери ще трябва малко повече да си поиграем, но много помага да си мислим за това което бихме искали да направим, как би станало върху дървото на извод и дали тази информация може хубаво да се кодира в правилата.

3.6 Няколко интересни задачи

Тук ще разгледаме няколко хубави задачи за безконтекстни граматики. Първата ще има малко по-нестандартно решение.

Твърдение 3.6.1. *Езикът* $L = \{\alpha \in \Sigma^* \mid |\alpha|_a = |\alpha|_b\}$ *е безконтекстен.*

Доказателство. Граматиката за L е следната:

$$S \to aSb \mid bSa \mid SS \mid \varepsilon$$

Сега нека покажем, че $\mathcal{L}(G) = L$:

($\subseteq$) За тази посока ще покажем с индукция по $n \in \mathbb{N}$, че ако $S \overset{n}{\lhd} \alpha$ и $\alpha \in \Sigma^*$, то $\alpha \in L$

- ако $S \overset{0}{\lhd} \alpha$, то $\alpha = S \notin \Sigma^*$ ✓
- ако $S \overset{n+1}{\lhd} \alpha$, тогава сме приложили някое правило

1 сл. Приложили сме правилото $S \to \varepsilon$. Тогава $\alpha = \varepsilon \in L$

2 сл. Приложили сме правилото $S \to aSb$. Тогава $S \overset{n}{\lhd} \beta$, като $\alpha = a\beta b$. Лесно се вижда, че имаме равенствата $|\alpha|_a = 1 + |\beta|_a \overset{\text{ИП}}{=} 1 + |\beta|_b = |\alpha|_b$, откъдето $\alpha \in L$

3 сл. Приложили сме правилото $S \to bSa$. Този случай е аналогичен на 2 сл.

4 сл. Приложили сме правилото $S \to SS$. Тогава $S \overset{n_1}{\lhd} \beta_1$ и $S \overset{n_2}{\lhd} \beta_2$, като $n = \max\{n_1, n_2\}$ и $\alpha = \beta_1\beta_2$. Така $|\alpha|_a = |\beta_1|_a + |\beta_2|_a \overset{\text{ИП}}{=} |\beta_1|_b + |\beta_2|_b = |\alpha|_b$, откъдето $\alpha \in L$

($\supseteq$) За тази посока ще покажем с индукция по $|\alpha|$, че ако $\alpha \in L$, то $S \overset{*}{\lhd} \alpha$

- $S \to \varepsilon$ е правило в G, и $S \overset{0}{\lhd} S$, откъдето $S \overset{1}{\lhd} \varepsilon$ ✓
- Нека $|\alpha| = 2n + 2$ (дължината очевидно трябва да е четна). Ако $\alpha = a\beta b$ за някое $\beta \in \Sigma^*$, то понеже $|\beta|_a = |\alpha|_a - 1 = |\alpha|_b - 1 = |\beta|_b$, по ИП $S \overset{*}{\lhd} \beta$. Но в G има правилото $S \to aSb$, откъдето $S \overset{*}{\lhd} \alpha$. Случаят, в който $\alpha = b\beta a$ е аналогичен. Интересните случаи са, когато $\alpha = x\beta x$ за $x \in \Sigma$. И двата са симетрични, така че ще разгледаме само когато $\alpha = a\beta a$. Нека $\alpha = \alpha_1 \dots \alpha_{2n+2}$, като $\alpha_i \in \Sigma$. Нека $f_\alpha(i) = |\alpha_1 \dots \alpha_i|_a - |\alpha_1 \dots \alpha_i|_b$. Например ако $\beta = aabb$, то $f_\beta(0) = 0, f_\beta(1) = 1, f_\beta(2) = 2, f_\beta(3) = 1$ и $f_\beta(4) = 0$. Ясно е, че понеже $\alpha \in L$, $f_\alpha(|\alpha|) = 0$. Понеже $\alpha_1 = a$, $f_\alpha(1) = 1$, и понеже $\alpha_{2n+2} = a$, $f_\alpha(2n+1) = -1$. Това, което можем да направим, е да продължим f_α до непрекъсната $g_\alpha : [0, 2n+2] \to \mathbb{R}$ по следния начин:
  - $g_\alpha(i) = f_\alpha(i)$ за всяко $i \in \{0, \dots, 2n+2\}$
  - $g_\alpha(i + \varepsilon) = (1 - \varepsilon)f_\alpha(i) + \varepsilon f_\alpha(i+1)$ за всяко $i \in 0, \dots, 2n+1$ и $\varepsilon \in (0, 1)$

  Очевидно g_α е непрекъсната в реалните (и не целите) си точки. Нека проверим, че е така и в целите:
  - $\lim\limits_{\varepsilon \to 0} g_\alpha(i + \varepsilon) = \lim\limits_{\varepsilon \to 0}[(1 - \varepsilon)f_\alpha(i) + \varepsilon f_\alpha(i+1)] = f_\alpha(i) = g_\alpha(i)$
  - $\lim\limits_{\varepsilon \to 1} g_\alpha(i + \varepsilon) = \lim\limits_{\varepsilon \to 1}[(1 - \varepsilon)f_\alpha(i) + \varepsilon f_\alpha(i+1)] = f_\alpha(i+1) = g_\alpha(i+1)$

  Имаме непрекъсната функция g_α, за която знаем, че $g_\alpha(1) = 1$ и $g_\alpha(2n+1) = -1$. Тогава от Теоремата на Болцано, има $i \in (1, 2n+1)$ такъв, че $g_\alpha(i) = 0$. Нещо повече, конструирали сме g така, че да връща цели стойности само когато и подаваме цели стойности. За това $i \in \{2, \dots, 2n\}$. Това означава, че $g_\alpha(i) = f_\alpha(i) = |\alpha_1 \dots \alpha_i|_a - |\alpha_1 \dots \alpha_i|_b = 0$, откъдето $\beta_1 = \alpha_1 \dots \alpha_i \in L$. Е тогава очевидно също и $\beta_2 = \alpha_{i+1} \dots \alpha_{2n+1} \in L$. По ИП $S \overset{*}{\lhd} \beta_1$ и $S \overset{*}{\lhd} \beta_2$. Така понеже имаме правилото $S \to SS$, $S \overset{*}{\lhd} \beta_1\beta_2 = \alpha$.

□

Дефиниция 3.6.2. Ще наричаме една дума α **добре скобуван относно** [**и**], ако за всяко $\beta \preceq_{pref} \alpha$, $|\beta|_[\geq |\beta|_]$ и $|\alpha|_[= |\alpha|_]$ Ще бележим този факт с $bal(\alpha, [,])$.

Пример за добра скобуван израз относно (и) би бил думата (((23+4)×(21-12))+6)/(73-23). Илюстрирано по-добре:

$$\underset{1}{(}\ \underset{2}{(}\ \underset{3}{(}\ 23 + 4\ \underset{2}{)}\ \times\ \underset{3}{(}\ 21 - 12\ \underset{2}{)}\ \underset{1}{)} + 6\ \underset{0}{)}\ /\ \underset{1}{(}\ 73 - 23\ \underset{0}{)}$$

Тук много лесно се вижда, че никога броят на (не надвишава броя на).

Задача 3.6.3. *Да се построи граматика за езика* $L = \{\alpha \in \{[,]\} \mid bal(a, [,])\}$
Упътване: много подобно на Твърдение 3.6.1

Твърдение 3.6.4. *Ако L е безконтекстен, тогава* $\mathrm{Pref}(L)$ *също е безконтекстен.*

Доказателство. Нека $G = \langle \Sigma, V, S, R \rangle$ е граматика в НФЧ без безполезни променливи за L. Под безполезни променливи се има предвид такива, които не генерират нищо от Σ^*.
Строим граматика $G_{pref} = \langle \Sigma, V_{pref}, \overleftarrow{S}, R_{pref} \rangle$ за $\mathrm{Pref}(L)$:

- $V_{pref} = V \cup \overleftarrow{V}$, където $\overleftarrow{V} = \{\overleftarrow{A} \mid A \in V\}$ и $V \cap \overleftarrow{V} = \varnothing$
- правилата от старата граматика се запазват
- ако $A \to_G BC$, то $\overleftarrow{A} \to_{G_{pref}} B\overleftarrow{C} \mid \overleftarrow{B}$
- ако $A \to_G a$, то $\overleftarrow{A} \to_{G_{pref}} a \mid \varepsilon$

Цялата идея на конструкцията е, че $\mathrm{Pref}(L_1 \cdot L_2) = \mathrm{Pref}(L_1) \cup (L_1 \cdot \mathrm{Pref}(L_2))$ (от свойства на Pref, Suff, Infix).
За да покажем, че $\mathrm{Pref}(L) \subseteq \mathcal{L}(G_{pref})$, ще докажем, че за всяко $A \in V$, $\alpha \in \Sigma^*$:

$$\text{ако } A \overset{*}{\lhd}_G \alpha, \text{ то } \overrightarrow{A} \overset{*}{\lhd}_{G_{pref}} \beta \text{ за всяко } \beta \preceq_{pref} \alpha \text{ т.е. } \mathrm{Pref}(\mathcal{L}_G(A)) \subseteq \mathcal{L}_{G_{pref}}(\overrightarrow{A})$$

Доказваме с индукция по големината на извода:

- ако $A \overset{0}{\lhd} \alpha$, то $\alpha = A \notin \Sigma^*$ ✓
- ако $A \overset{n+1}{\lhd} \alpha$, то сме приложили някое правило

  1 сл. Приложили сме правилото $A \to_G BC$. Тогава $B \overset{n_1}{\lhd} \beta_1$ и $C \overset{n_1}{\lhd} \beta_2$, като $n = \max\{n_1, n_2\}$ и $\alpha = \beta_1\beta_2$. Нека $\gamma \preceq_{pref} \alpha$. Тогава има два варианта (от свойства на Pref, Suff, Infix):
  - $\gamma \preceq_{pref} \beta_1$ - тогава по ИП $\overrightarrow{B} \overset{*}{\lhd} \gamma$, и понеже имаме правилото $\overrightarrow{A} \to \overrightarrow{B}$, $\overrightarrow{A} \overset{*}{\lhd} \gamma$.
  - $\gamma = \beta_1\lambda$ и $\lambda \preceq_{pref} \beta_2$ - тогава по ИП $\overrightarrow{C} \overset{*}{\lhd} \lambda$, и понеже имаме правилото $\overrightarrow{A} \to B\overrightarrow{C}$, $\overrightarrow{A} \overset{*}{\lhd} \beta_1\lambda = \gamma$.

  2 сл. Приложили сме правилото $A \to_G a$. Тогава $\alpha = a$ и всичките префикси на α са a и ε. Тъй като $A \to_G a$, $\overrightarrow{A} \to_{G_{pref}} a \mid \varepsilon$, откъдето $\overrightarrow{A} \overset{*}{\lhd} a$ и $\overrightarrow{A} \overset{*}{\lhd} \varepsilon$

За да покажем, че $\mathcal{L}(G_{pref}) \subseteq \mathrm{Pref}(L)$, ще докажем, че за всяко $A \in V$, $\alpha \in \Sigma^*$:

$$\text{ако } \overrightarrow{A} \overset{*}{\lhd}_{G_{pref}} \alpha, \text{ то има } \beta \in \Sigma^*, \text{ че } A \overset{*}{\lhd}_G \alpha\beta \text{ т.е. } \mathcal{L}_{G_{pref}}(\overrightarrow{A}) \subseteq \mathrm{Pref}(\mathcal{L}_G(A))$$

Доказваме с индукция по големината на извода:

- ако $\overrightarrow{A} \overset{0}{\lhd} \alpha$, то $\alpha = \overrightarrow{A} \notin \Sigma^*$ ✓
- ако $\overrightarrow{A} \overset{n+1}{\lhd} \alpha$, то сме приложили някое правило

  1 сл. Приложили сме правилото $\overrightarrow{A} \to_{G_{pref}} B\overrightarrow{C}$. Тогава $B \overset{n_1}{\lhd}_{G_{pref}} \beta_1$ $\overrightarrow{C}_{G_{pref}} \overset{n_2}{\lhd} \beta_2$ като $n = \max\{n_1, n_2\}$ и $\alpha = \beta_1\beta_2$. По ИП има $\lambda \in \Sigma^*$ такова, че $C \overset{*}{\lhd}_G \beta_2\lambda$. От правилото, което сме приложили знаем, че $A \to_G BC$, откъдето $A \overset{*}{\lhd} \beta_1\beta_2\lambda = \alpha\lambda$.

  2 сл. Приложили сме правилото $\overrightarrow{A} \to_G \overrightarrow{B}$. Тогава $\overrightarrow{B} \overset{n}{\lhd}_{G_{pref}} \alpha$ и по ИП има $\beta_1 \in \Sigma^*$ такова, че $B \overset{*}{\lhd}_G \alpha\beta_1$. От правилото, което сме приложили знаем, че $A \to BC$. Тъй като C не е безполезна променлива, има $\beta_2 \in \Sigma^*$ такова, че $C \overset{*}{\lhd}_G \beta_2$. Тогава $A \overset{*}{\lhd}_G \alpha\beta_1\beta_2$.

  3 сл. Приложили сме правилото $\overrightarrow{A} \to_{G_{pref}} a$. Тогава $A \to_G a$, откъдето $A \overset{*}{\lhd}_G a = a\varepsilon$.

  4 сл. Приложили сме правилото $\overrightarrow{A} \to_{G_{pref}} \varepsilon$. Тогава $A \to_G a$, откъдето $A \overset{*}{\lhd}_G a = \varepsilon a$.

Накрая завършваме с:

$$\mathrm{Pref}(L) = \mathrm{Pref}(\mathcal{L}(G)) = \mathrm{Pref}(\mathcal{L}_G(S)) = \mathcal{L}_{G_{pref}}(\overrightarrow{S}) = \mathcal{L}(G_{pref})$$

□

Задача 3.6.5. *Да се покаже, че ако L е безконтекстен, то тогава* $\mathrm{Suff}(L)$ *и* $\mathrm{Infix}(L)$ *също са безконтекстни. Упътване: конструкция за* Suff *е много подобна, а тази за* Infix *е комбинация от двете; може и да се използват свойства на Pref, Suff, Infix*

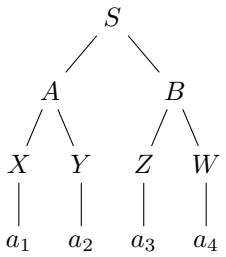
3.7 Сечение на безконтекстен език с регулярен

Сега ще разгледаме една много хубава операция, с едно нейно хубаво приложение. Въпреки че не можем да сечем два безконтекстни езика с надеждите винаги да получим безконтекстен език, можем да заслабим малко това условие.

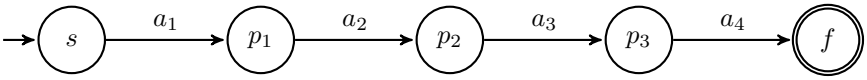
Твърдение 3.7.1. *За всеки безконтекстен език L и регулярен език M, езикът $L \cap M$ е безконтекстен.*

Доказателство. Нека $G = \langle \Sigma, V, R, S \rangle$ е граматика в НФЧ за L и нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е ДКА за M. Да си представим някаква дума $\alpha = a_1a_2a_3a_4 \in L \cap M$. Някои неща които знаем за нея:

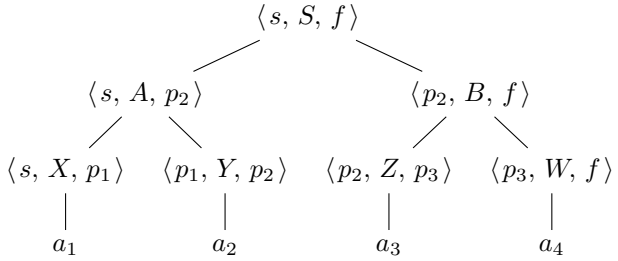
- има дърво на извод за α, съгласувано с G:



- в $\mathcal{A}$ има път с етикет α от s до някое $f \in F$ т.е. $\delta^*(s, \alpha) = f \in F$:



Това, което ще направим, е да се опитаме да кодираме информацията за автомата в дърветата на извод:



Новите променливи в граматиката ще са от вида $\langle p, A, q \rangle$ като идеята е да се генерират всички думи α такива, че:

$$A \overset{*}{\lhd} \alpha \text{ и } \delta^*(p, \alpha) = q$$

Нека опишем сега формално конструкцията за новата ни граматика $G' = \langle \Sigma, V', S', R' \rangle$:

- $V' = (Q \times V \times Q) \cup \{S'\}$, където $S' \notin Q \times V \times Q$
- за всички $A \in V$ и $p, q, r \in Q$, ако $A \to_G BC$, то $\langle p, A, q \rangle \to_{G'} \langle p, B, r \rangle \langle r, C, q \rangle$
  Тук граматиката се опитва да "познае" думата, която генерира какъв път ще измине в автомата. Предварително тя няма как да го знае, тъй като самата граматика не знае каква дума ще генерира.
- за всички $A \in V$, $p, q \in Q$ и $a \in \Sigma$, ако $A \to_G a$ и $\delta(p, a) = q$, то $\langle p, A, q \rangle \to_{G'} a$
  Тук вече става истинската "синхронизация". Ако граматиката е познала преходите на думата, която е решила да генерира, то тя вече наистина ще я генерира. Ако не е познала, понеже не са правилни или няма такива, то няма да се генерира нищо.
- за всяко $f \in F$, $S' \to_{G'} \langle s, S, f \rangle$

Сега трябва да покажем, че за всички $A \in V$ и $p, q \in Q$:

$$\mathcal{L}_{G'}(\langle p, A, q \rangle) = \mathcal{L}_G(A) \cap \{\alpha \in \Sigma^* \mid \delta^*(p, \alpha) = q\}$$

Този надпис хубаво илюстрира как разделяме голямото сечение на L и M на по-малки сечения, което е идеята на конструкцията.

Твърдение 3.7.2. *За всички $A \in V$, $p, q \in Q$, $n \in \mathbb{N}$ и $\alpha \in \Sigma^*$:*

$$\langle p,\, A,\, q\rangle \overset{n}{\lhd}_{G'} \alpha \iff A \overset{n}{\lhd}_{G} \alpha \,\&\, \delta^*(p, \alpha) = q$$

Доказателство. С индукция по $n \in \mathbb{N}$.

- базата е тривиално изпълнена и в двете посоки ✓
- ще покажем ИС и в двете посоки:

(⇒) Нека $\langle p,\, A,\, q\rangle \overset{n+1}{\lhd}_{G'} \alpha$. Тогава сме приложили някое правило.

1 сл. Приложили сме правилото $\langle p,\, A,\, q\rangle \to_{G'} a$. Тогава $\alpha = a$, $\delta(p, a) = q$ и $A \to_G a$, откъдето $A \overset{1}{\lhd} \alpha$.

2 сл. Приложили сме правилото $\langle p,\, A,\, q\rangle \to_{G'} \langle p,\, B,\, r\rangle\langle r,\, C,\, q\rangle$. Тогава съществуват $\alpha_1, \alpha_2 \in \Sigma^*$ такива, че $\langle p,\, B,\, r\rangle \overset{n_1}{\lhd}_{G'} \alpha_1$ и $\langle r,\, C,\, q\rangle \overset{n_2}{\lhd}_{G'} \alpha_2$ като $\alpha = \alpha_1\alpha_2$ и $n = \max\{n_1, n_2\}$. Също така $A \to BC$.
По ИП:
· $B \overset{n_1}{\lhd}_{G} \alpha_1$ и $\delta^*(p, \alpha_1) = r$
· $C \overset{n_2}{\lhd}_{G} \alpha_2$ и $\delta^*(r, \alpha_2) = q$

Така можем да заключим, че $A \overset{n+1}{\lhd}_{G} \alpha_1\alpha_2 = \alpha$ и $\delta^*(p, \alpha) = \delta^*(p, \alpha_1\alpha_2) = \delta^*(\delta^*(p, \alpha_1), \alpha_2) = q$

(⇐) Нека $A \overset{n+1}{\lhd}_{G} \alpha$ и $\delta^*(p, \alpha) = q$. Тогава сме приложили някое правило.

1 сл. Приложили сме правилото $A \to_G a$. Тогава $\alpha = a$ и понеже $\delta(p, a) = q$, $\langle p,\, A,\, q\rangle \to_{G'} a$, откъдето получаваме, че $\langle p,\, A,\, q\rangle \overset{1}{\lhd} \alpha$.

2 сл. Приложили сме правилото $A \to BC$. Тогава съществуват $\alpha_1, \alpha_2 \in \Sigma^*$ такива, че $B \overset{n_1}{\lhd}_{G} \alpha_1$ и $C \overset{n_2}{\lhd}_{G} \alpha_2$ като $\alpha = \alpha_1\alpha_2$ и $n = \max\{n_1, n_2\}$. Нека $r = \delta^*(p, \alpha_1)$. Очевидно $\delta^*(r, \alpha_2) = q$. Също така имаме правилото $\langle p,\, A,\, q\rangle \to_{G'} \langle p,\, B,\, r\rangle\langle r,\, C,\, q\rangle$.
По ИП:
· $\langle p,\, B,\, r\rangle \overset{n_1}{\lhd} \alpha_1$
· $\langle r,\, C,\, q\rangle \overset{n_2}{\lhd} \alpha_2$

Така можем да заключим, че $\langle p,\, A,\, q\rangle \overset{n+1}{\lhd}_{G'} \alpha$

□

Имайки всичко това, остава само да завършим с:

$$\begin{aligned}
\alpha \in \mathcal{L}(G') &\iff \alpha \in \mathcal{L}_{G'}(S') \\
&\iff \alpha \in \bigcup_{f \in F} \mathcal{L}_{G'}(\langle s,\, S,\, f\rangle) \\
&\iff \alpha \in \bigcup_{f \in F} (\mathcal{L}_G(S) \cap \{\alpha \in \Sigma^* \mid \delta^*(s, \alpha) = f\}) \\
&\iff \alpha \in \mathcal{L}(G) \cap \bigcup_{f \in F} \{\alpha \in \Sigma^* \mid \delta^*(s, \alpha) = f\} \\
&\iff \alpha \in \mathcal{L}(G) \cap \mathcal{L}(\mathcal{A}) \\
&\iff \alpha \in L \cap M
\end{aligned}$$

□

Следствие 3.7.3. *Ако $L \cap M$ не е безконтекстен език и M е регулярен език, то L не е безконтекстен.*

Доказателство. Ако L беше безконтекстен, то тогава $L \cap M$ щеше също да бъде безконтекстен, имайки че M е регулярен език. □

Твърдение 3.7.4. *Езикът $L = \{\alpha \in \Sigma^* \mid |\alpha|_a = |\alpha|_b = |\alpha|_c\}$ не е безконтекстен.*

Доказателство. Знаем, че езикът $L_0 = \{a^n b^n c^n \mid n \in \mathbb{N}\}$ не е безконтекстен. Очевидно $L_0 = L \cap (\{a\}^*\{b\}^*\{c\}^*)$. От Следствие 3.7.3 получаваме, че L не е безконтекстен. □

3.8 Стекови автомати

Сега ще видим как можем да си мислим за безконтекстните граматики по друг, "по-познат" начин.

Дефиниция 3.8.1. Стеков автомат ще наричаме всяко $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f \rangle$, където:

- Σ е крайна входна азбука
- Γ е крайна стекова азбука
- $\sharp \in \Gamma$ е символ за дъното на стека
- Q е крайно множество от състояния
- $s \in Q$ е начално състояние
- $\Delta : Q \times \Sigma_\varepsilon \times \Gamma \to \mathcal{P}(Q \times \Gamma^{\leq 2})$ е функция на преходите
- $f \in Q$ е финално състояние

За един преход $\langle q, \beta \rangle \in \Delta(p, x, A)$ можем да си мислим по следния начин:

четейки x (което е буква или ε) и намирайки се в състоянието p,
ако на върха на стека е буквата A,
можем да отидем в състоянието q като буквата A се маха от върха на стека и се заменя с β

Дефиниция 3.8.2. Конфигурация в стеков автомат $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f \rangle$ ще наричаме всяка тройка от вида $\langle q, \alpha, \gamma \rangle \in Q \times \Sigma^* \times \Gamma^*$.

Можем да си мислим за $\langle q, \alpha, \gamma \rangle$ по следния начин:

в момента се намираме в състояние q, в стека е думата γ, и остава да прочетем α

Дефиниция 3.8.3. Въвеждаме релацията $\vdash_P$ за едностъпков преход между конфигурации по следното правило:

ако $\langle q, \beta \rangle \in \Delta(p, x, A)$, то $\langle p, x\alpha, A\gamma \rangle \vdash_P \langle q, \alpha, \beta\gamma \rangle$

Също така въвеждаме релацията $\vdash_P^l$ за многостъпков преход между конфигурации по следното правило:

- $\kappa \vdash_P^0 \kappa$
- ако $\kappa_1 \vdash_P \kappa_2$ и $\kappa_2 \vdash_P^l \kappa_3$, то $\kappa_1 \vdash_P^{l+1} \kappa_3$

Казваме, че $\kappa_1 \vdash_P^* \kappa_2$ има $l \in \mathbb{N}$ такова, че $\kappa_1 \vdash_P^l \kappa_2$.

Забележка. Ако P се подразбира няма да го пишем.

Дефиниция 3.8.4. Езика на стеков автомат $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f \rangle$ ще бележим с

$$\mathcal{L}(P) = \{\alpha \in \Sigma^* \mid \langle s, \alpha, \sharp \rangle \vdash_P^* \langle f, \varepsilon, \varepsilon \rangle\}$$

Сега ще покажем някои полезни свойства на $\vdash^*$.

Твърдение 3.8.5. *Ако $\kappa_1 \vdash^{l_1} \kappa_2$ и $\kappa_2 \vdash^{l_2} \kappa_3$, то $\kappa_1 \vdash^{l_1+l_2} \kappa_3$*

Доказателство. С индукция по l_1:

- Нека $\kappa_1 \vdash^0 \kappa_2$ и $\kappa_2 \vdash^l \kappa_3$, тогава $\kappa_1 = \kappa_2$ ✓
- Нека $\kappa_1 \vdash^{l_1+1} \kappa_2$ и $\kappa_2 \vdash^{l_2} \kappa_3$. Тогава $\kappa_1 \vdash \kappa'$ и $\kappa' \vdash^{l_1} \kappa_2$. По ИП $\kappa' \vdash^{l_1+l_2} \kappa_3$, откъдето $\kappa_1 \vdash^{1+l_1+l_2} \kappa_3$.

□

Твърдение 3.8.6. *Ако $\langle p, \alpha_1, \gamma_1 \rangle \vdash^l \langle q, \varepsilon, \varepsilon \rangle$, то $\langle p, \alpha_1\alpha_2, \gamma_1\gamma_2 \rangle \vdash^l \langle q, \alpha_2, \gamma_2 \rangle$.*

Доказателство. С индукция по l:

- Ако $\langle p, \alpha_1, \gamma_1 \rangle \vdash^0 \langle q, \varepsilon, \varepsilon \rangle$, то $p = q$, $\alpha_1 = \varepsilon$ и $\gamma_1 = \varepsilon$. Тогава очевидно $\langle p, \alpha_1\alpha_2, \gamma_1\gamma_2 \rangle \vdash^0 \langle q, \alpha_2, \gamma_2 \rangle$ ✓
- Ако $\langle p, x\alpha_1, A\gamma_1 \rangle \vdash^{l+1} \langle q, \varepsilon, \varepsilon \rangle$, то $\langle p, x\alpha_1, A\gamma_1 \rangle \vdash \langle r, \alpha_1, \beta\gamma_1 \rangle \vdash^l \langle q, \varepsilon, \varepsilon \rangle$, където $\langle r, \beta \rangle \in \Delta(p, x, A)$. По ИП $\langle r, \alpha_1\alpha_2, \beta\gamma_1\gamma_2 \rangle \vdash^l \langle q, \alpha_2, \gamma_2 \rangle$, откъдето $\langle p, x\alpha_1\alpha_2, A\gamma_1\gamma_2 \rangle \vdash^{l+1} \langle q, \alpha_2, \gamma_2 \rangle$.

□

Следствие 3.8.7. *Ако* $\langle p, \alpha_1, A_1 \rangle \vdash^{l_1} \langle r, \varepsilon, \varepsilon \rangle$ *и* $\langle r, \alpha_2, A_2 \rangle \vdash^{l_2} \langle q, \varepsilon, \varepsilon \rangle$, *то* $\langle p, \alpha_1\alpha_2, A_1A_2 \rangle \vdash^{l_1+l_2} \langle q, \varepsilon, \varepsilon \rangle$

Твърдение 3.8.8. *Нека* $\langle p, \alpha, A\gamma \rangle \vdash^{l} \langle q, \varepsilon, \varepsilon \rangle$. *Тогава има* $l_1, l_2 \in \mathbb{N}$, *състояние* r *и думи* α_1, α_2 *такива, че:*

- $l = l_1 + l_2$
- $\alpha = \alpha_1\alpha_2$
- $\langle p, \alpha_1, A \rangle \vdash^{l_1} \langle r, \varepsilon, \varepsilon \rangle$
- $\langle r, \alpha_2, \gamma \rangle \vdash^{l_2} \langle q, \varepsilon, \varepsilon \rangle$

Доказателство. С индукция по l (очевидно $l \geq 1$):

- Ако $\langle p, \alpha, A\gamma \rangle \vdash^{1} \langle q, \varepsilon, \varepsilon \rangle$, то $\gamma = \varepsilon$, $\alpha \in \Sigma_\varepsilon$ и $\langle q, \varepsilon \rangle \in \Delta(p, \alpha, A)$. Полагаме $l_1 = 1$, $l_2 = 0$, $\alpha_1 = \alpha$, $\alpha_2 = \varepsilon$, $r = q$ и сме готови ✓
- Нека $\langle p, \alpha, A\gamma \rangle \vdash^{l+1} \langle q, \varepsilon, \varepsilon \rangle$. Тогава сме направили поне един преход. Нека $\alpha = x\lambda$.

  1 сл. На първа стъпка сме имали прехода $\langle p', BC \rangle \in \Delta(p, x, A)$. Тогава $\langle p', \lambda, BC\gamma \rangle \vdash^{l} \langle q, \varepsilon, \varepsilon \rangle$. По ИП:

  * $l'_1 + l'_2 = l$
  * $\lambda_1\lambda_2 = \lambda$
  * $\langle p', \lambda_1, B \rangle \vdash^{l'_1} \langle r', \varepsilon, \varepsilon \rangle$
  * $\langle r', \lambda_2, C\gamma \rangle \vdash^{l'_2} \langle q, \varepsilon, \varepsilon \rangle$

  Понеже $l'_2 \leq l$ пак да приложим ИП:

  * $k_1 + k_2 = l_2$
  * $\beta_1\beta_2 = \lambda_2$
  * $\langle r', \beta_1, C \rangle \vdash^{k_1} \langle r'', \varepsilon, \varepsilon \rangle$
  * $\langle r'', \beta_2, \gamma \rangle \vdash^{k_2} \langle q, \varepsilon, \varepsilon \rangle$

  Полагаме $l_1 = 1 + l'_1 + k_1$, $l_2 = k_2$, $r = r''$, $\alpha_1 = x\lambda_1\beta_1$, $\alpha_2 = \beta_2$ и получаваме, че:

  * $l_1 + l_2 = 1 + l'_1 + k_1 + k_2 = 1 + l'_1 + l'_2 = 1 + l$
  * $\alpha_1\alpha_2 = x\lambda_1\beta_1\beta_2 = x\lambda_1\lambda_2 = x\lambda = \alpha$
  * $\langle p, x\lambda_1\beta_1, A \rangle \vdash \langle p', \lambda_1\beta_1, BC \rangle \vdash^{l'_1} \langle r', \beta_1, C \rangle \vdash^{k_1} \langle r'', \varepsilon, \varepsilon \rangle$, откъдето $\langle p, \alpha_1, A \rangle \vdash^{l_1} \langle r, \varepsilon, \varepsilon \rangle$
  * $\langle r'', \beta_2, \gamma \rangle \vdash^{k_2} \langle q, \varepsilon, \varepsilon \rangle$, откъдето $\langle r, \alpha_2, \gamma \rangle \vdash^{l_2} \langle q, \varepsilon, \varepsilon \rangle$

  2 сл. На първа стъпка сме имали прехода $\langle p', B \rangle \in \Delta(p, x, A)$. Тогава $\langle p', \lambda, B\gamma \rangle \vdash^{l} \langle q, \varepsilon, \varepsilon \rangle$. По ИП:

  * $l'_1 + l'_2 = l$
  * $\lambda_1\lambda_2 = \lambda$
  * $\langle p', \lambda_1, B \rangle \vdash^{l'_1} \langle r', \varepsilon, \varepsilon \rangle$
  * $\langle r', \lambda_2, \gamma \rangle \vdash^{l'_2} \langle q, \varepsilon, \varepsilon \rangle$

  Полагаме $l_1 = 1 + l'_1$, $l_2 = l'_2$, $r = r'$, $\alpha_1 = x\lambda_1$, $\alpha_2 = \lambda_2$ и получаваме, че:

  * $l_1 + l_2 = 1 + l'_1 + l'_2 = 1 + l$
  * $\alpha_1\alpha_2 = x\lambda_1\lambda_2 = x\lambda = \alpha$
  * $\langle p, x\lambda_1, A \rangle \vdash \langle p', \lambda_1, B \rangle \vdash^{l'_1} \langle r', \varepsilon, \varepsilon \rangle$, откъдето $\langle p, \alpha_1, A \rangle \vdash^{l_1} \langle r, \varepsilon, \varepsilon \rangle$
  * $\langle r', \lambda_2, \gamma \rangle \vdash^{l'_2} \langle q, \varepsilon, \varepsilon \rangle$, откъдето $\langle r, \alpha_2, \gamma \rangle \vdash^{l_2} \langle q, \varepsilon, \varepsilon \rangle$

  3 сл. На първа стъпка сме имали прехода $\langle p', \varepsilon \rangle \in \Delta(p, x, A)$. Тогава $\langle p', \lambda, \gamma \rangle \vdash^{l} \langle q, \varepsilon, \varepsilon \rangle$. Тогава полагаме $l_1 = 1$, $l_2 = l$, $r = p'$, $\alpha_1 = x, \alpha_2 = \lambda$ и получаваме, че:

  * $l_1 + l_2 = 1 + l$
  * $\alpha_1\alpha_2 = x\lambda = \alpha$
  * $\langle p, x, A \rangle \vdash \langle p', \varepsilon, \varepsilon \rangle$, откъдето $\langle p, \alpha_1, A \rangle \vdash^{l_1} \langle r, \varepsilon, \varepsilon \rangle$
  * $\langle p', \lambda, \gamma \rangle \vdash^{l} \langle q, \varepsilon, \varepsilon \rangle$, откъдето $\langle r, \alpha_2, \gamma \rangle \vdash^{l_2} \langle q, \varepsilon, \varepsilon \rangle$

С това доказателството е завършено. □

3.9 Класически примери за стекови автомати

Тук ще покажем някои прости езици, които се разпознават от стекови автомати.

Твърдение 3.9.1. *Езикът $L = \{a^n b^n \mid n \in \mathbb{N}\}$ се разпознава от стеков автомат.*

Доказателство. Стековият автомат $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f \rangle$ е следния:

- $Q = \{s, p, f\}$ и $\Gamma = \{\sharp, a\}$
- $\Delta(s, \varepsilon, \sharp) = \{\langle f, \varepsilon \rangle\}$ - директно разпознаваме ε
- $\Delta(s, a, \sharp) = \{\langle s, a\sharp \rangle\}$ - трупаме a в стека
- $\Delta(s, a, a) = \{\langle s, aa \rangle\}$ - трупаме a в стека
- $\Delta(s, b, a) = \{\langle p, \varepsilon \rangle\}$ - минаваме в режим на премахване от стека и четене само на b
- $\Delta(p, b, a) = \{\langle p, \varepsilon \rangle\}$ - премахваме натрупаните a в стека
- $\Delta(p, \varepsilon, \sharp) = \{\langle f, \varepsilon \rangle\}$ - когато няма нищо в стека a и b са били срещати равен брой пъти
- Във всички останали случаи функцията Δ връща $\varnothing$ (по нататък няма да го пишем)

За да покажем, че $L \subseteq \mathcal{L}(P)$, трябва да се покаже (оставяме на читателя), че за всяко $n \in \mathbb{N}$:

- $\langle s, a^n\beta, \sharp \rangle \vdash^n \langle s, \beta, a^n\sharp \rangle$
  Това просто казва, че състоянието s се използва за да трупа a в стека.
- $\langle p, b^n, a^n\sharp \rangle \vdash^n \langle p, \varepsilon, \sharp \rangle$
  Това просто казва, че състоянието p се използва за да премахне натрупаните b в стека.

Имайки това $\langle s, a^n b^n, \sharp \rangle \vdash^n \langle s, b^n, a^n\sharp \rangle \vdash^n \langle p, \varepsilon, \sharp \rangle \vdash \langle f, \varepsilon, \varepsilon \rangle$, откъдето $a^n b^n \in \mathcal{L}(P)$.
За да покажем, че $\mathcal{L}(P) \subseteq L$, трябва да се покаже (оставяме на читателя), че за всяко $n \in \mathbb{N}$:

- ако $\langle s, \alpha\beta, \sharp \rangle \vdash^n \langle s, \beta, \gamma\sharp \rangle$, то $\alpha = \gamma = a^n$
  Тук трябва да се използва, че от s няма преходи с други букви, които да остават в s.
- ако $\langle p, \beta, \gamma\sharp \rangle \vdash^n \langle p, \varepsilon, \sharp \rangle$, то $\beta = b^n$ и $\gamma = a^n$
  Тук трябва да се използва, че от p няма преходи с други букви, които да остават в p.

Нека $\alpha \in \mathcal{L}(P)$. Тогава $\langle s, \alpha, \sharp \rangle \vdash^n \langle f, \varepsilon, \varepsilon \rangle$. Ясно е, че $\varepsilon \in L$, така че нека Б.О.О. $\alpha \neq \varepsilon$. Тогава знаем, че сме минали през всички състояния. Нека $\alpha_1, \alpha_2 \in \Sigma^*$ са такива, че $\alpha = \alpha_1\alpha_2$ и

$$\underbrace{\langle s, \alpha_1\alpha_2, \sharp \rangle \vdash^{n_1} \langle p, \alpha_2, \gamma\sharp \rangle}_{\substack{\text{трябва да направим такъв преход} \\ \text{четейки дума, различна от } \varepsilon}} \vdash^{n_2} \underbrace{\langle p, \varepsilon, \sharp \rangle \vdash \langle f, \varepsilon, \varepsilon \rangle}_{\substack{\text{единственият начин} \\ \text{да разпознаем дума} \\ \text{различна от } \varepsilon}}$$

Тогава от горното твърдение $\alpha_1 = \gamma = a^{n_1}$, $\alpha_2 = b^{n_2}$ и $\gamma = a^{n_2}$. Така $\alpha = a^{n_1}b^{n_1} = a^{n_2}b^{n_2} \in L$. □

Задача 3.9.2. *Да се построят стекови автомати за следните езици:*

- $L_1 = \{a^n b^{2n} \mid n \in \mathbb{N}\}$
- $L_2 = \{a^{2n} b^n \mid n \in \mathbb{N}\}$
- $L_3 = \{a^{2n} b^{3n} \mid n \in \mathbb{N}\}$

Упътване: не трябва да се правят много промени, трябва само да се промени малко работата със стека

Твърдение 3.9.3. *Съществува стеков автомат за езика $L = \{\alpha \in \Sigma^* \mid |\alpha|_a = |\alpha|_b\}$.*

Доказателство. Стековият автомат $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f \rangle$ е следния (доказателството остава за читателя):

- $Q = \{s, f\}$ и $\Gamma = \{a, b, \sharp\}$
- $\Delta(s, x, \sharp) = \{\langle s, x\sharp \rangle\}$ за $x \in \Sigma$ - при празен стек добавяме буквата, която е в повече
- $\Delta(s, x, x) = \{\langle s, xx \rangle\}$ за $x \in \Sigma$ - продължаваме да добавяме буквата в повече
- $\Delta(s, a, b) = \Delta(s, b, a) = \{\langle s, \varepsilon \rangle\}$ - при срещане на другата буква махаме от стека
- $\Delta(s, \varepsilon, \sharp) = \{\langle f, \varepsilon \rangle\}$ - ако стека е празен броят на различните букви е равен

За да се докаже, че $L = \mathcal{L}(P)$ можем да покажем, че за всяка дума γ и за всяко естествено n:

- $a^n\gamma \in L \iff \langle s, \gamma, a^n\sharp\rangle \vdash^* \langle s, \varepsilon, \sharp\rangle$
- $b^n\gamma \in L \iff \langle s, \gamma, b^n\sharp\rangle \vdash^* \langle s, \varepsilon, \sharp\rangle$

Трябва да се направи индукция по $|\gamma|$, също така двете части се показват едновременно.
Накрая заключаваме, че:

$$\alpha \in L \iff \langle s, \alpha, a^0\sharp\rangle \vdash^* \langle s, \varepsilon, \sharp\rangle \vdash \langle f, \varepsilon, \varepsilon\rangle \iff \alpha \in \mathcal{L}(P)$$

□

Една често срещана задача за стек, е да се провери дали един израз е "добре скобуван" (Дефиниция 3.6.2).

Твърдение 3.9.4. *Езикът $L = \{\alpha \in \Sigma^* \mid bal(\alpha, a, b)\}$ се разпознава от стеков автомат.*

Доказателство. Стековият автомат $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f\rangle$ е следния (доказателството остава за читателя):

- $Q = \{s, f\}$ и $\Gamma = \{a, \sharp\}$
- $\Delta(s, a, \sharp) = \{\langle s, a\sharp\rangle\}$ - добавяме "отварящите скоби" в стека
- $\Delta(s, a, a) = \{\langle s, aa\rangle\}$ - добавяме "отварящите скоби" в стека
- $\Delta(s, b, a) = \{\langle s, \varepsilon\rangle\}$ - махаме "отварящите скоби" от стека като четем "затварящи"
- $\Delta(s, \varepsilon, \sharp) = \{\langle f, \varepsilon\rangle\}$ - когато стека е празен, прочетената дума до сега е балансирана

За да се докаже, че $L = \mathcal{L}(P)$ можем да покажем, че за всяка дума γ и за всяко естествено n:

$$a^n\gamma \in L \iff \langle s, \gamma, a^n\sharp\rangle \vdash^* \langle s, \varepsilon, \sharp\rangle$$

Доказателството върви по същият начин като в Твърдение 3.9.3.
Накрая заключаваме, че:

$$\alpha \in L \iff \langle s, \alpha, a^0\sharp\rangle \vdash^* \langle s, \varepsilon, \sharp\rangle \vdash \langle f, \varepsilon, \varepsilon\rangle \iff \alpha \in \mathcal{L}(P)$$

□

Задача 3.9.5. *Да се направи стеков автомат за $L = \{\alpha \in \{a, b, c, d\}^* \mid bal(\alpha, a, b)\ \&\ bal(\alpha, c, d)\}$.*
Упътване: напълно аналогично на Твърдение 3.9.4

3.10 Еквивалентност на безконтекстните граматики и стековите автомати

Както видяхме има езици, които се разпознават и от стекови автомати, и от безконтекстни граматики. Оказва се, че всеки език, който се разпознава от безконтекстна граматика се разпознава от стеков автомат и обратното.

Лема 3.10.1. *За всяка безконтекстна граматика G съществува стеков автомат P с $\mathcal{L}(P) = \mathcal{L}(G)$.*

Доказателство. Б.О.О. нека $G = \langle \Sigma, V, S, R\rangle$ е граматика в НФЧ.
Строим стеков автомат $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f\rangle$ за $\mathcal{L}(G)$:

- $Q = \{s, p, f\}$
- $\Gamma = \Sigma \cup V \cup \{\sharp\}$
- $\Delta(s, \varepsilon, \sharp) = \{\langle p, S\sharp\rangle\}$ - в началото започваме с началната променлива в стека
- $\Delta(p, \varepsilon, A) = \{\langle p, \alpha\rangle \mid A \to_G \alpha \in R\}$ за всяко $A \in V$ - симулираме генерирането на променливи
- $\Delta(p, x, x) = \langle p, \varepsilon\rangle$ за всяко $x \in \Sigma$ - симулираме генерирането на букви
- $\Delta(p, \varepsilon, \sharp) = \{\langle f, \varepsilon\rangle\}$ - когато сме няма променливи в стека и няма букви за четене завършваме работа

Сега ще покажем, че за всяко $A \in V$ и $\alpha \in \Sigma^*$:

$$A \overset{*}{\lhd} \alpha \iff \langle p, \alpha, A\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$$

И за двете посоки правим индукция по $n \in \mathbb{N}$. Базите са тривиално изпълнени. Ще опишем подробно ИС.

($\Rightarrow$) Нека $A \overset{n+1}{\lhd} \alpha$. Тогава сме приложили някое правило:

1 сл. Приложили сме правило $A \to a$. Тогава $\alpha = a$. Понеже имаме такова правило можем да направим прехода $\langle p, \varepsilon a, A\rangle \vdash \langle p, a, a\rangle \vdash \langle p, \varepsilon, \varepsilon\rangle$.

2 сл. Приложили сме правило $A \to BC$. Тогава $B \overset{n_1}{\lhd} \alpha_1$ и $C \overset{n_2}{\lhd} \alpha_2$ като $\alpha = \alpha_1\alpha_2$ и $n = \max\{n_1, n_2\}$. По ИП имаме, че $\langle p, \alpha_1, B\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$ и $\langle p, \alpha_2, C\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$, откъдето заради Следствие 3.8.7 можем да направим прехода $\langle p, \varepsilon\alpha_1\alpha_2, A\rangle \vdash \langle p, \alpha_1\alpha_2, BC\rangle \vdash^* \langle p, \alpha_2, C\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$.

($\Leftarrow$) Нека $\langle p, \alpha, A\rangle \vdash^{n+1} \langle p, \varepsilon, \varepsilon\rangle$. Тогава имаме следните два варианта:

1 сл. Направили сме преходите $\langle p, \varepsilon\alpha, A\rangle \vdash \langle p, \alpha, a\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$. Тогава имаме правилото $A \to a$ и $\alpha = a$. Очевидно $A \overset{*}{\lhd} a$.

2 сл. Направили сме преходите $\langle p, \varepsilon\alpha, A\rangle \vdash \langle p, \alpha, BC\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$. Тогава имаме правилото $A \to BC$ и по Твърдение 3.8.8 има α_1 и α_2 такива, че $\alpha = \alpha_1\alpha_2$, $\langle p, \alpha_1, B\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$ и $\langle p, \alpha_2, C\rangle \vdash^* \langle p, \varepsilon, \varepsilon\rangle$. По ИП $B \overset{*}{\lhd} \alpha_1$ и $C \overset{*}{\lhd} \alpha_2$, откъдето $A \overset{*}{\lhd} \alpha_1\alpha_2 = \alpha$.

Накрая получаваме, че:

$$\begin{aligned}\alpha \in \mathcal{L}(G) &\iff S \overset{*}{\lhd} \alpha \\ &\iff \langle s, \varepsilon\alpha, \sharp\rangle \vdash \langle p, \alpha, S\sharp\rangle \vdash^* \langle p, \varepsilon, \sharp\rangle \vdash \langle f, \varepsilon, \varepsilon\rangle \\ &\iff \alpha \in \mathcal{L}(P)\end{aligned}$$

□

Лема 3.10.2. *За всеки стеков автомат P съществува безконтекстна граматика G с $\mathcal{L}(G) = \mathcal{L}(P)$.*

Доказателство. Нека $P = \langle \Sigma, \Gamma, \sharp, Q, s, \Delta, f\rangle$ е стеков автомат.
Строим $G = \langle \Sigma, V, S, R\rangle$ за $\mathcal{L}(P)$:

- $V = Q \times \Gamma \times Q$
- $S = \langle s, \sharp, f\rangle$
- ако $\langle q, \varepsilon\rangle \in \Delta(p, x, A)$, то имаме правилото $\langle p, A, q\rangle \to_G x$
- ако $\langle r, B\rangle \in \Delta(p, x, A)$, то имаме правилото $\langle p, A, q\rangle \to_G x\langle r, B, q\rangle$ за всяко $q \in Q$
- ако $\langle r, BC\rangle \in \Delta(p, x, A)$, то имаме правилото $\langle p, A, q\rangle \to_G x\langle r, B, r'\rangle\langle r', C, q\rangle$ за всяко $q, r' \in Q$

Сега трябва да покажем, че за всяко $p, q \in Q$, $A \in \Gamma$ и $\alpha \in \Sigma^*$:

$$\langle p, A, q\rangle \overset{*}{\lhd} \alpha \iff \langle p, \alpha, A\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle.$$

И за двете посоки правим индукция по $n \in \mathbb{N}$. Базите са тривиално изпълнени. Ще опишем подробно ИС.

($\Rightarrow$) Нека $\langle p, A, q\rangle \overset{n+1}{\lhd} \alpha$. Тогава сме приложили някое правило:

1 сл. Приложили сме правилото $\langle p, A, q\rangle \to x$. Тогава $\alpha = x$, откъдето по конструкция $\langle p, x, A\rangle \vdash \langle q, \varepsilon, \varepsilon\rangle$.

2 сл. Приложили сме правилото $\langle p, A, q\rangle \to x\langle r, B, q\rangle$. Тогава $\alpha = x\lambda$ и $\langle r, B, q\rangle \overset{n}{\lhd} \lambda$, откъдето можем да видим, че $\langle p, x\lambda, A\rangle \vdash \langle r, \lambda, B\rangle \overset{\text{ИП}}{\vdash^*} \langle q, \varepsilon, \varepsilon\rangle$.

3 сл. Приложили сме правилото $\langle p, A, q\rangle \to x\langle r, B, r'\rangle\langle r', C, q\rangle$. Тогава $\langle r, B, r'\rangle \overset{n_1}{\lhd} \lambda_1$ и $\langle r', C, q\rangle \overset{n_2}{\lhd} \lambda_2$ като $\alpha = x\lambda_1\lambda_2$ и $n = \max\{n_1, n_2\}$. По ИП $\langle r, \lambda_1, B\rangle \vdash^* \langle r', \varepsilon, \varepsilon\rangle$ и $\langle r', \lambda_2, C\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle$, откъдето по Следствие 3.8.7 и конструкция на граматиката $\langle p, x\lambda_1\lambda_2, A\rangle \vdash \langle r, \lambda_1\lambda_2, BC\rangle \vdash^* \langle r', \lambda_2, C\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle$.

($\Leftarrow$) Нека $\langle p, x\lambda, A\rangle \vdash^{n+1} \langle q, \varepsilon, \varepsilon\rangle$. Тогава имаме следните три варианта:

1 сл. Направили сме преходите $\langle p, x\lambda, A\rangle \vdash \langle r, \lambda, \varepsilon\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle$. Тогава очевидно $\lambda = \varepsilon$ и $r = q$, откъдето в G имаме правилото $\langle p, A, q\rangle \to x$. Така $\langle p, A, q\rangle \overset{*}{\lhd} x = x\lambda$.

2 сл. Направили сме преходите $\langle p, x\lambda, A\rangle \vdash \langle r, \lambda, B\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle$. Тогава имаме правилото $\langle p, A, q\rangle \to x\langle r, B, q\rangle$. По ИП $\langle r, B, q\rangle \overset{*}{\lhd} \lambda$, откъдето $\langle p, A, q\rangle \overset{*}{\lhd} x\lambda$.

3 сл. Направили сме преходите $\langle p, x\lambda, A\rangle \vdash \langle r, \lambda, BC\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle$. Тогава по Твърдение 3.8.8 има $\lambda_1, \lambda_2 \in \Sigma^*$ и $r' \in Q$ такива, че $\lambda = \lambda_1\lambda_2$, $\langle r, \lambda_1, B\rangle \vdash^* \langle r', \varepsilon, \varepsilon\rangle$ и $\langle r', \lambda_2, C\rangle \vdash^* \langle q, \varepsilon, \varepsilon\rangle$. Също така имаме правилото $\langle p, A, q\rangle \to x\langle r, B, r'\rangle\langle r', C, q\rangle$. По ИП $\langle r, B, r'\rangle \overset{*}{\lhd} \lambda_1$ и $\langle r', C, q\rangle \overset{*}{\lhd} \lambda_2$, откъдето $\langle p, A, q\rangle \overset{*}{\lhd} x\lambda_1\lambda_2 = x\lambda$.

Накрая получаваме, че:

$$\begin{aligned}\alpha \in \mathcal{L}(G) &\iff \langle s, \sharp, f\rangle \overset{*}{\lhd} \alpha \\ &\iff \langle s, \alpha, \sharp\rangle \vdash^* \langle f, \varepsilon, \varepsilon\rangle \\ &\iff \alpha \in \mathcal{L}(P)\end{aligned}$$

□

3.11 Задачи за упражнение

Задача 3.11.1. *Да се докаже, че сечението с регулярен език запазва безконтекстност изцяло с автомати.*
Упътване: конструкцията е много подобна на тази с детерминираните автомати

Задача 3.11.2. *Да се докаже, че за произволен хомоморфизъм (Дефиниция 2.15.1) h и безконтекстен език L, езикът $h[L]$ е безконтекстен.*
Упътване: да се вземе граматика в НФЧ и да се помисли как трябва да се променят дърветата на извод

Задача 3.11.3. *Използвайки, че езикът $\{a^n b^n \mid n \in \mathbb{N}\}$ е безконтекстен, да се докаже, че следните езици са безконтекстни:*

- $L_1 = \{b^n a^n \mid n \in \mathbb{N}\}$
- $L_2 = \{b^{2n} a^n \mid n \in \mathbb{N}\}$

Упътване: да се представят езиците като хомоморфизми върху L

Задача 3.11.4. *Нека L е регулярен. Използвайки, че е безконтекстен (Твърдение 3.2.6) езикът*

$$\mathrm{Move}(L) = \{a^n b^m \mid (\exists \alpha \in L)\ (|\alpha|_a = n \,\&\, |\alpha|_b = m)\}$$

да се докаже, че следните езици са безконтекстни:

- $L' = \{\alpha_1 \dots \alpha_n \beta_1 \dots \beta_m \mid (\exists \alpha \in L)\ (|\alpha|_a = n \,\&\, |\alpha|_b = m) \,\&\, \alpha_i \in L_1 \,\&\, \beta_i \in L_2\}$
- $L'' = \{\alpha_1 \dots \alpha_{2n} \beta_1 \dots \beta_{3m} \mid (\exists \alpha \in L)\ (|\alpha|_a = n \,\&\, |\alpha|_b = m) \,\&\, \alpha_i \in L_1 \,\&\, \beta_i \in L_2\}$
- $L''' = \{\beta_1 \dots \beta_m \alpha_1 \dots \alpha_n \mid (\exists \alpha \in L)\ (|\alpha|_a = n \,\&\, |\alpha|_b = m) \,\&\, \alpha_i \in L_1 \,\&\, \beta_i \in L_2\}$

Упътване: да се представят езиците като хомоморфизми или композиция от хомоморфизми върху $\mathrm{Move}(L)$

Дефиниция 3.11.5. Функция $h : \Sigma_1^* \to \mathcal{P}(\Sigma_2^*)$ ще наричаме **безконтекстен хомоморфизъм**, ако:

- $h(x)$ е безконтекстен за всяко $x \in \Sigma_1$
- $h(\alpha \cdot \beta) = h(\alpha) \cdot h(\beta)$ за всички $\alpha, \beta \in \Sigma_1^*$

Задача 3.11.6. *Да се докаже, че за произволен безконтекстен хомоморфизъм h е вярно, че $h(\varepsilon) = \{\varepsilon\}$.*

Задача 3.11.7. *Да се докаже, че за произволен безконтекстен хомоморфизъм h и безконтекстен език L, езикът $\bigcup h[L]$ е безконтекстен.*
Упътване: същата конструкцията като Задача 3.11.2

Задача 3.11.8. *Да се покаже, че $L = \{a^n b^k c^n d^k \mid n, k \in \mathbb{N}\}$ не е безконтекстен.*

Задача 3.11.9. *Да се покаже, че операцията $\mathrm{Cyc}_{rev}(L) = \{\alpha\beta^{rev} \in \Sigma^* \mid \beta\alpha \in L\}$ не запазва безконтекстност.*
Упътване: използвайте Следствие 3.7.3 като пресечате $\{x_1\}^*\{x_2\}^*\{x_3\}^*\{x_4\}^*$ с някой безконтекстен език, подобен на езика от Задача 3.11.8.