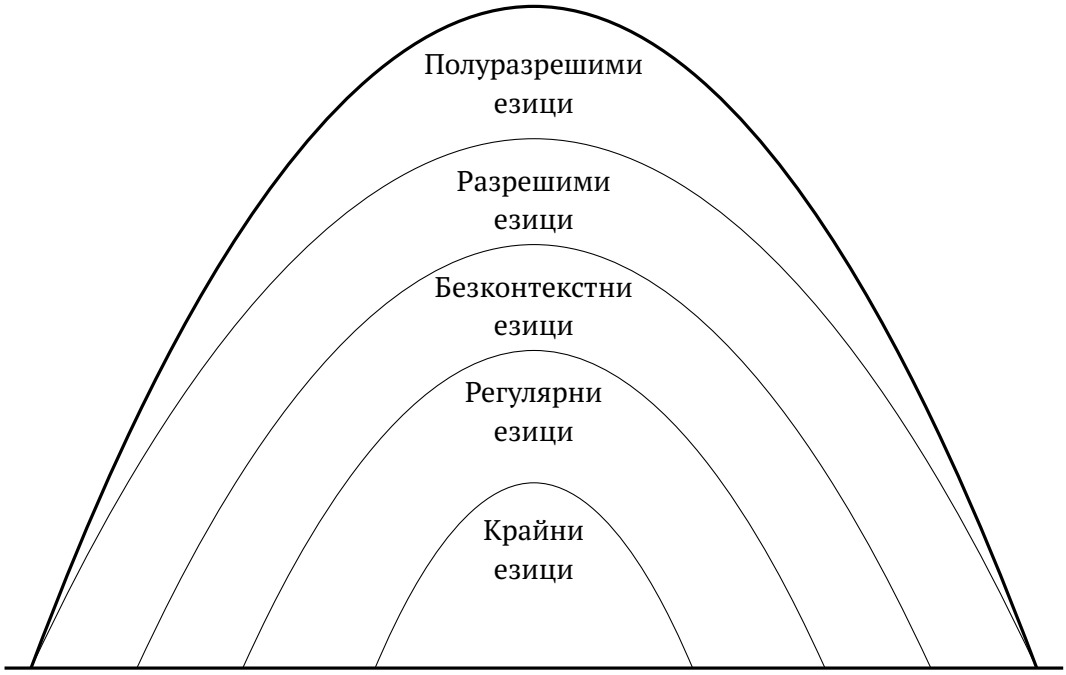
Записки по „Езици, автомати, изчислимост”

Стефан Вътев[1]

10 януари 2022 г.

[1]Това е чернова. Възможни са неточности и грешки, а също и несъответсвия с терминологията въведена на лекции. За забележки и коментари: stefanv@fmi.uni-sofia.bg


Полуразрешими
езици
Разрешими
езици
Безконтекстни
езици
Регулярни
езици
Крайни
езици

Съдържание

Глава 1

Увод

1.1 Съждително смятане

Както при езиците за програмиране, всяка логика има свой синтаксис и семантика. Тук ще разгледаме класическата съждителна логика, при която те са сравнително прости.

На англ. Propositional calculus

Съждителното смятане наподобява аритметичното смятане, като вместо аритметичните операции $+, -, \cdot, /$, имаме съждителни операции като $\neg, \wedge, \vee$. Например, $(p \vee q) \wedge \neg r$ е съждителна формула. Освен това, докато аритметичните променливи приемат стойности числа, то съждителните променливи приемат само стойности **истина (1)** или **неистина (0)**.

Съждителна формула наричаме съвкупността от съждителни променливи $p, q, r, \ldots$, свързани със знаците за логически операции $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ и скоби, определящи реда на операциите.

Това не е формална дефиниция, но за момента е достатъчно.

Съждителни операции

- Отрицание $\neg$
- Дизюнкция $\vee$
- Конюнкция $\wedge$
- Импликация $\rightarrow$
- Еквивалентност $\Leftrightarrow$

Ще използваме таблица за истинност за да определим стойностите на основните съждителни операции при всички възможни набори на стойностите на променливите.

p	q	$\neg p$	$p \vee q$	$p \wedge q$	$p \to q$	$\neg p \vee q$	$p \Leftrightarrow q$	$(p \wedge q) \vee (\neg p \wedge \neg q)$
0	0	1	0	0	1	1	1	1
0	1	1	1	0	1	1	0	0
1	0	0	1	0	0	0	0	0
1	1	0	1	1	1	1	1	1

Съждително верен (валиден) е този логически израз, който има верностна стойност **1** при всички възможни набори на стойностите на съждителните променливи в израза, т.е. стълбът на израза в таблицата за истинност трябва да съдържа само стойности **1**.

Два съждителни израза φ и ψ са **еквивалентни**, което означаваме $\varphi \equiv \psi$, ако са съставени от едни и същи съждителни променливи и двата израза имат едни и същи верностни стойности при всички комбинации от верностни стойности на променливите. С други думи, колоните на двата израза в съответните им таблици за истинност трябва да съвпадат. Така например, от горната таблица се вижда, че $p \to q \equiv \neg p \vee q$ и $p \Leftrightarrow q \equiv (\neg p \wedge q) \vee (p \wedge \neg q)$.

1.1.1 Съждителни закони

I) **Закон за идемпотентността**

$$p \wedge p \equiv p$$

$$p \vee p \equiv p$$

II) **Комутативен закон**

$$p \vee q \equiv q \vee p$$

$$p \wedge q \equiv q \wedge p$$

III) **Асоциативен закон**

$$(p \vee q) \vee r \equiv p \vee (q \vee r)$$

$$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$$

IV) **Дистрибутивен закон**

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$$

$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

V) **Закони на де Морган**

$$\neg(p \wedge q) \equiv (\neg p \vee \neg q)$$

$$\neg(p \vee q) \equiv (\neg p \wedge \neg q)$$

VI) **Закон за контрапозицията**

$$p \to q \equiv \neg q \to \neg p$$

VII) **Обобщен закон за контрапозицията**

$$(p \wedge q) \to r \equiv (p \wedge \neg r) \to \neg q$$

VIII) **Закон за изключеното трето**

$$p \vee \neg p \equiv \mathbf{1}$$

IX) **Закон за силогизма (транзитивност)**

$$((p \to q) \ \wedge \ (q \to r)) \to (p \to r) \equiv \mathbf{1}$$

Лесно се проверява с таблиците за истинност, че законите са валидни.

1.2 Предикати и квантори

Квантори

Свойствата или отношенията на елементите в произволно множество се наричат **предикати**. Нека да разгледаме един едноместен предикат $P(\cdot)$.

твърдение	Кога е истина?	Кога е неистина?
$\forall x P(x)$	$P(x)$ е вярно за всяко x	съществува x, за което $P(x)$ **не** е вярно
$\exists x P(x)$	съществува x, за което $P(x)$ е вярно	$P(x)$ **не** е вярно за всяко x

(I) **Квантор за общност** $\forall x$. Записът $(\forall x \in A)P(x)$ означава, че за всеки елемент a в A, твърдението $P(a)$ има стойност истина. Например, $(\forall x \in \mathbb{R})[(x+1)^2 = x^2 + 2x + 1]$.

(II) **Квантор за съществуване** $\exists x$. Записът $(\exists x \in A)P(x)$ означава, че съществува елемент a в A, за който твърдението $P(a)$ има стойност истина. Например, $(\exists x \in \mathbb{C})[x^2 = -1]$, но $(\forall x \in \mathbb{R})[x^2 \neq -1]$.

Закони на предикатното смятане

(I) $\neg\forall x P(x) \Leftrightarrow \exists x \neg P(x)$

(II) $\neg\exists x P(x) \Leftrightarrow \forall x \neg P(x)$

(III) $\forall x P(x) \Leftrightarrow \neg\exists x \neg P(x)$

(IV) $\exists x P(x) \Leftrightarrow \neg\forall x \neg P(x)$

(V) $\forall x \forall y P(x,y) \Leftrightarrow \forall y \forall x P(x,y)$

(VI) $\exists x \exists y P(x,y) \Leftrightarrow \exists y \exists x P(x,y)$

(VII) $\exists x \forall y P(x,y) \rightarrow \forall y \exists x P(x,y)$

Закони на Де Морган за квантори			
Твърдение	Еквивалентно твърдение	Кога е истина?	Кога е неистина?
$\neg\exists x P(x)$	$\forall x \neg P(x)$	за всяко x $P(x)$ **не** е вярно	съществува x, за което $P(x)$ е вярно
$\neg\forall x P(x)$	$\exists x \neg P(x)$	съществува x, за което $P(x)$ **не** е вярно	$P(x)$ е вярно за всяко x

Задача 1.1. Да означим с $K(x,y)$ твърдението "x познава y". Изразете като предикатна формула следните твърдения.

1) Всеки познава някого. $\forall x \exists y K(x,y)$

2) Някой познава всеки. $\exists x \forall y K(x,y)$

3) Някой е познаван от всички. $\exists x \forall y K(y,x)$

4) Всеки знае някой, който не го познава. $\forall x \exists y (K(x,y) \wedge \neg K(y,x))$

5) Има такъв, който знае всеки, който го познава. $\exists x \forall y (K(y,x) \rightarrow K(x,y))$

6) Всеки двама познати имат общ познат. $(\forall x,y)(K(x,y)\,\&\,K(y,x) \rightarrow \exists z(K(x,z)\,\&\,K(y,z))$

Пример 1.1. Нека $D \subseteq \mathbb{R}$. Казваме, че $f : D \rightarrow \mathbb{R}$ е *непрекъсната* в точката $x_0 \in D$, ако

$$(\forall \varepsilon > 0)(\exists \delta > 0)(\forall x \in D)(\ |x_0 - x| < \delta \ \rightarrow \ |f(x_0) - f(x)| < \varepsilon\ ).$$

Да видим какво означава f да бъде *прекъсната* в точката $x_0 \in D$: f е прекъсната в x_0 точно тогава, когато f не е непрекъсната в x_0

$$\begin{aligned}
&\neg(\forall \varepsilon > 0)(\exists \delta > 0)(\forall x \in D)(\ |x_0 - x| < \delta \ \rightarrow \ |f(x_0) - f(x)| < \varepsilon\ ) \equiv \\
&(\exists \varepsilon > 0)\neg(\exists \delta > 0)(\forall x \in D)(\ |x_0 - x| < \delta \ \rightarrow \ |f(x_0) - f(x)| < \varepsilon\ ) \equiv \\
&(\exists \varepsilon > 0)(\forall \delta > 0)\neg(\forall x \in D)(\ |x_0 - x| < \delta \ \rightarrow \ |f(x_0) - f(x)| < \varepsilon\ ) \equiv \\
&(\exists \varepsilon > 0)(\forall \delta > 0)(\exists x \in D)\neg(\ |x_0 - x| < \delta \ \rightarrow \ |f(x_0) - f(x)| < \varepsilon\ ) \equiv \\
&(\exists \varepsilon > 0)(\forall \delta > 0)(\exists x \in D)\neg(\ \neg(|x_0 - x| < \delta) \vee |f(x_0) - f(x)| < \varepsilon\ ) \equiv \\
&(\exists \varepsilon > 0)(\forall \delta > 0)(\exists x \in D)(\ \neg\neg(|x_0 - x| < \delta) \wedge \neg(|f(x_0) - f(x)| < \varepsilon)\ ) \equiv \\
&(\exists \varepsilon > 0)(\forall \delta > 0)(\exists x \in D)(\ |x_0 - x| < \delta \ \wedge \ |f(x_0) - f(x)| \geq \varepsilon\ ).
\end{aligned}$$

1.3 Множества, релации, функции

Основни отношения между множества

За произволни множества A и B, ще казваме, че:

- A е подмножество на B, което ще означаваме като $A \subseteq B$, ако:

$$(\forall x)[x \in A \implies x \in B].$$

- A е равно на B, което ще означаваме като $A = B$, ако:

$$(\forall x)[x \in A \Leftrightarrow x \in B],$$

или

$$A = B \Leftrightarrow A \subseteq B \,\&\, B \subseteq A.$$

Основни операции върху множества

Ще разгледаме няколко операции върху произволни множества A и B.

На англ. *intersection*

- **Сечение**

$$A \cap B = \{x \mid x \in A \wedge x \in B\}.$$

Казано по-формално, $A \cap B$ е множеството, за което е изпълнено, че:

$$(\forall x)[x \in A \cap B \Leftrightarrow (x \in A \wedge x \in B)].$$

Примери:

- $A \cap A = A$, за всяко множество A.
- $A \cap \emptyset = \emptyset$, за всяко множество A.
- $\{1, \emptyset, \{\emptyset\}\} \cap \{\emptyset\} = \{\emptyset\}$.
- $\{1, 2, \{1, 2\}\} \cap \{1, \{1\}\} = \{1\}$.

Макар и $\emptyset$, $\{\emptyset\}$ и $\{1, 2\}$ да са множества, те може да са елементи на други множества.

На англ. *union*

- **Обединение**

$$A \cup B = \{x \mid x \in A \vee x \in B\}.$$

$A \cup B$ е множеството, за което е изпълнено, че:

$$(\forall x)[x \in A \cup B \Leftrightarrow (x \in A \vee x \in B)].$$

Примери:

- $A \cup A = A$, за всяко множество A.
- $A \cup \emptyset = A$, за всяко множество A.
- $\{1, 2, \emptyset\} \cup \{1, 2, \{\emptyset\}\} = \{1, 2, \emptyset, \{\emptyset\}\}$.

– $\{1, 2, \{1, 2\}\} \cup \{1, \{1\}\} = \{1, 2, \{1\}, \{1, 2\}\}$.

- **Разлика**

$$A \setminus B = \{x \mid x \in A \wedge x \notin B\}.$$

$A \setminus B$ е множеството, за което е изпълнено, че:

$$(\forall x)[x \in A \setminus B \Leftrightarrow (x \in A \wedge x \notin B)].$$

Примери:

– $A \setminus A = \emptyset$, за всяко множество A.
– $A \setminus \emptyset = A$, за всяко множество A.
– $\emptyset \setminus A = \emptyset$, за всяко множество A.
– $\{1, 2, \emptyset\} \setminus \{1, 2, \{\emptyset\}\} = \{\emptyset\}$.
– $\{1, 2, \{1, 2\}\} \setminus \{1, \{1\}\} = \{2, \{1, 2\}\}$.

- **Степенно множество**

$$\mathscr{P}(A) = \{x \mid x \subseteq A\}.$$

$\mathscr{P}(A)$ е множеството, за което е изпълнено, че:

На англ. *power set*

$$(\forall x)[x \in \mathscr{P}(A) \Leftrightarrow (\forall y)[y \in x \to y \in A]].$$

Примери:

В литературата се среща също така и означението 2^A за степенното множество на A.

– $\mathscr{P}(\emptyset) = \{\emptyset\}$.
– $\mathscr{P}(\{\emptyset\}) = \{\emptyset, \{\emptyset\}\}$.
– $\mathscr{P}(\{\emptyset, \{\emptyset\}\}) = \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}, \{\emptyset, \{\emptyset\}\}\}$.
– $\mathscr{P}(\{1, 2\}) = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$.

Задача 1.2. Проверете верни ли са свойствата:

а) $A \subseteq B \Leftrightarrow A \setminus B = \emptyset \Leftrightarrow A \cup B = B \Leftrightarrow A \cap B = A$;

б) $A \setminus \emptyset = A$, $\emptyset \setminus A = \emptyset$, $A \setminus B = B \setminus A$.

в) $A \cap (B \cup A) = A \cap B$;

г) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ и $A \cap (B \cup C) = (A \cup B) \cap (A \cup C)$;

д) $A \setminus B = A \Leftrightarrow A \cap B = \emptyset$;

е) $A \setminus B = A \setminus (A \cap B)$ и $A \setminus B = A \setminus (A \cup B)$;

ж) $(A \cup B) \setminus C = (A \setminus C) \cup (B \setminus C)$;

з) $C \setminus (A \cup B) = (C \setminus A) \cap (C \setminus B)$ и $C \setminus (A \cap B) = (C \setminus A) \cup (C \setminus B)$

Закони на Де Морган

и) $(A\backslash B)\backslash C = (A\backslash C)\backslash(B\backslash C)$ и $A\backslash(B\backslash C) = (A\backslash B) \cup (A \cap C)$;

к) $A \subseteq B \Rightarrow \mathscr{P}(A) \subseteq \mathscr{P}(B)$;

$X \subseteq A \cup B \overset{?}{\Rightarrow} X \subseteq A \vee X \subseteq B$

л) $\mathscr{P}(A \cap B) = \mathscr{P}(A) \cap \mathscr{P}(B)$ и $\mathscr{P}(A \cup B) = \mathscr{P}(A) \cup \mathscr{P}(B)$;

За да дадем определение на понятието релация, трябва първо да въведем понятието декартово произведение на множества, което пък от своя страна се основава на понятието наредена двойка.

Наредена двойка

За два елемента a и b въвеждаме опрецията **наредена двойка** $\langle a, b\rangle$. Наредената двойка $\langle a, b\rangle$ има следното характеристичното свойство:

$$a_1 = a_2 \ \wedge \ b_1 = b_2 \ \Leftrightarrow \ \langle a_1, b_1\rangle = \langle a_2, b_2\rangle.$$

Понятието наредена двойка може да се дефинира по много начини, стига да изпълнява харектеристичното свойство. Ето примери как това може да стане:

Norbert Wiener (1914)

1) Първото теоретико-множествено определение на понятието наредена двойка е дадено от Норберт Винер:

$$\langle a, b\rangle \overset{\text{деф}}{=} \{\{\{a\}, \emptyset\}, \{\{b\}\}\}.$$

Kazimierz Kuratowski (1921)

2) Определението на Куратовски се приема за „стандартно” в наши дни:

$$\langle a, b\rangle \overset{\text{деф}}{=} \{\{a\}, \{a, b\}\}.$$

Задача 1.3. Докажете, че горните дефиниции наистина изпълняват характеристичното свойство за наредени двойки.

Пример за индуктивна (рекурсивна) дефиниция

Определение 1.1. Сега можем, за всяко естествено число $n \geq 1$, да въведем понятието наредена n-орка $\langle a_1, \ldots, a_n\rangle$:

$$\langle a_1\rangle \overset{\text{деф}}{=} a_1,$$
$$\langle a_1, a_2, \ldots, a_n\rangle \overset{\text{деф}}{=} \langle a_1, \langle a_2, \ldots, a_n\rangle\rangle.$$

Оттук нататък ще считаме, че имаме дадено понятието наредена n-орка, без да се интересуваме от нейната формална дефиниция.

Декартово произведение

На англ. cartesian product. Считаме, че $(A \times B) \times C = A \times (B \times C) = A \times B \times C$.

За две множества A и B, определяме тяхното декартово произведение като

$$A \times B = \{\langle a, b\rangle \mid a \in A \,\&\, b \in B\}.$$

За краен брой множества $A_1, A_2, \ldots, A_n$, определяме

$$A_1 \times A_2 \times \cdots \times A_n = \{\langle a_1, a_2, \ldots, a_n\rangle \mid a_1 \in A_1 \,\&\, \ldots \,\&\, a_n \in A_n\}.$$

Задача 1.4. Проверете, че:

а) $A \times (B \cup C) = (A \times B) \cup (A \times C)$.

б) $(A \cup B) \times C = (A \times C) \cup (B \times C)$.

в) $A \times (B \cap C) = (A \times B) \cap (A \times C)$.

г) $(A \cap B) \times C = (A \times C) \cap (B \times C)$.

д) $A \times (B \setminus C) = (A \times B) \setminus (A \times C)$.

е) $(A \setminus B) \times C = (A \times C) \setminus (B \times C)$.

Видове функции

Функцията $f : A \to B$ е:

- **инекция**, ако е изпълнено свойството: // или f е **обратима**

$$(\forall a_1, a_2 \in A)[\, a_1 \neq a_2 \;\to\; f(a_1) \neq f(a_2)\,],$$

  или еквивалентно,

$$(\forall a_1, a_2 \in A)[\, f(a_1) = f(a_2) \;\to\; a_1 = a_2\,].$$

- **сюрекция**, ако е изпълнено свойството: // или f е **върху** B

$$(\forall b \in B)(\exists a \in A)[\, f(a) = b\,].$$

- **биекция**, ако е инекция и сюрекция.

Задача 1.5. Докажете, че $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ е биекция, където Канторово кодиране. Най-добре се вижда като се нарисува таблица

$$f(x, y) = \frac{(x + y)(x + y + 1)}{2} + x.$$

Основни видове бинарни релации

Подмножествата R от вида $R \subseteq A \times A \times \cdots \times A$ се наричат релации. Релациите от вида $R \subseteq A \times A$ са важен клас, който ще срещаме често. Да разгледаме няколко основни видове релации от този клас:

I) **рефлексивна**, ако

$$(\forall x \in A)[\langle x, x\rangle \in R].$$

Например, бинарната релация $\leq$ над $\mathbb{N}$ е рефлексивна, защото

$$(\forall x \in \mathbb{N})[\, x \leq x\,].$$

II) **транзитивна**, ако

$$(\forall x, y, z \in A)[\, \langle x, y\rangle \in R \,\&\, \langle y, z\rangle \in R \to \langle x, z\rangle \in R\,].$$

Например, бинарната релация $\leq$ над $\mathbb{N}$ е транзитивна, защото

$$(\forall x, y, z \in \mathbb{N})[\, x \leq y \,\&\, y \leq z \;\to\; x \leq z\,].$$

III) **симетрична**, ако

$$(\forall x, y \in A)[\, \langle x, y\rangle \in R \to \langle y, x\rangle \in R\,].$$

Например, бинарната релация $=$ над $\mathbb{N}$ е рефлексивна, защото

$$(\forall x, y \in \mathbb{N})[\, x = y \;\to\; y = x\,].$$

IV) **антисиметрична**, ако

$$(\forall x, y \in A)[\, \langle x, y\rangle \in R \,\&\, \langle y, x\rangle \in R \to x = y\,].$$

Например, релацията $\leq \subseteq\ \mathbb{N} \times \mathbb{N}$ е антисиметрична, защото

$$(\forall x, y, z \in A)[\, x \leq y \,\&\, y \leq x \;\to\; x = y\,].$$

- Една бинарна релация R над множеството A се нарича **релация на еквивалентност**, ако R е рефлексивна, транзитивна и симетрична.

- За всеки елемент $a \in A$, определяме неговия **клас на еквивалентност** относно релацията на еквивалентност R по следния начин:

$$[a]_R \stackrel{\text{деф}}{=} \{b \in A \mid \langle a, b\rangle \in R\}.$$

Забележка. Лесно се съобразява, че за всеки два елемента $a, b \in A$,

$$\langle a, b\rangle \in R \;\Leftrightarrow\; [a]_R = [b]_R.$$

Пример 1.2. За всяко естествено число $n \geq 2$, дефинираме релацията R_n като

$$\langle x, y\rangle \in R_n \;\Leftrightarrow\; x \equiv y \,(\mathrm{mod}\ n).$$

Ясно е, че R_n са релации на еквивалентност.

Операции върху бинарни релации

I) **Композиция** на две релации $R \subseteq B \times C$ и $P \subseteq A \times B$ е релацията $R \circ P \subseteq A \times C$, определена като:

Това е малко объркващо

$$R \circ P \stackrel{\text{деф}}{=} \{\langle a, c\rangle \in A \times C \mid (\exists b \in B)[\langle a, b\rangle \in P \,\&\, \langle b, c\rangle \in R]\}.$$

II) **Рефлексивно затваряне** на релацията $R \subseteq A \times A$ е релацията

Очевидно е, че P е рефлексивна релация, дори ако R не е.

$$P \stackrel{\text{деф}}{=} R \cup \{\langle a, a\rangle \mid a \in A\}.$$

III) **Итерация** на релацията $R \subseteq A \times A$ дефинираме като за всяко естествено число n, дефинираме релацията R^n по следния начин:

Лесно се вижда, че $R^1 = R$

$$R^0 \stackrel{\text{деф}}{=} \{\langle a, a\rangle \mid a \in A\}$$
$$R^{n+1} \stackrel{\text{деф}}{=} R^n \circ R.$$

IV) **Транзитивно затваряне** на $R \subseteq A \times A$ е релацията

✍ Проверете, че R^+ е транзитивна релация!

$$R^+ \stackrel{\text{деф}}{=} \bigcup_{n \geq 1} R^n.$$

За дадена релация R, с $R^\star$ ще означаваме нейното *рефлексивно и транзитивно затваряне*. От дефинициите е ясно, че

$$R^\star = \bigcup_{n \geq 0} R^n.$$

1.4 Доказателства на твърдения

Допускане на противното

Да приемем, че искаме да докажем, че свойството $P(x)$ е вярно за всяко естествено число. Един начин да направим това е следният:

- Допускаме, че съществува елемент n, за който $\neg P(n)$.
- Използвайки, че $\neg P(n)$ правим извод, от който следва факт, за който знаем, че винаги е лъжа. Това означава, че доказваме следното твърдение

$$\exists x \neg P(x) \to \mathbf{0}.$$

- Тогава можем да заключим, че $\forall x P(x)$, защото имаме следния извод:

$$\dfrac{\dfrac{\dfrac{\exists x \neg P(x) \to \mathbf{0}}{\mathbf{1} \to \neg\exists x \neg P(x)}}{\neg\exists x \neg P(x)}}{\forall x P(x)}$$

Ще илюстрираме този метод като решим няколко прости задачи.

Задача 1.6. За всяко $a \in \mathbb{Z}$, ако a^2 е четно, то a е четно.

Доказателство. Ние искаме да докажем твърдението P, където:

$$P \equiv (\forall a \in \mathbb{Z})[a^2 \text{ е четно } \to \ a \text{ е четно}].$$

$\neg(\forall x)(A(x) \to B(x))$ е еквивалентно на $(\exists x)(A(x) \wedge \neg B(x))$

Да допуснем противното, т.е. изпълнено е $\neg P$. Лесно се вижда, че

$$\neg P \ \Leftrightarrow \ (\exists a \in \mathbb{Z})[a^2 \text{ е четно } \wedge \ a \text{ не е четно}].$$

Да вземем едно такова нечетно a, за което a^2 е четно. Това означава, че $a = 2k + 1$, за някое $k \in \mathbb{Z}$, и

$$a^2 = (2k + 1)^2 = 4k^2 + 4k + 1,$$

което очевидно е нечетно число. Но ние допуснахме, че a^2 е четно. Така достигаме до противоречие, следователно нашето допускане е грешно и

$$(\forall a \in \mathbb{Z})[a^2 \text{ е четно } \to \ a \text{ е четно}].$$

□

Задача 1.7. Докажете, $\sqrt{2}$ **не** е рационално число.

Доказателство. Да допуснем, че $\sqrt{2}$ е рационално число. Тогава съществуват $a, b \in \mathbb{Z}$, такива че

$$\sqrt{2} = \frac{a}{b}.$$

Без ограничение, можем да приемем, че a и b са естествени числа, които нямат общи делители, т.е. не можем да съкратим дробта $\frac{a}{b}$. Получаваме, че

$$2b^2 = a^2.$$

Тогава a^2 е четно число и от Задача 1.6, a е четно число. Нека $a = 2k$, за някое естествено число k. Получаваме, че

$$2b^2 = 4k^2,$$

от което следва, че

$$b^2 = 2k^2.$$

Това означава, че b също е четно число, $b = 2n$, за някое естествено число n. Следователно, a и b са четни числа и имат общ делител 2, което е противоречие с нашето допускане, че a и b нямат общи делители. Така достигаме до противоречие. Накрая заключаваме, че $\sqrt{2}$ не е рационално число. □

Индукция върху естествените числа

Доказателството с индукция по $\mathbb{N}$ представлява следната схема:

Да напомним, че естествените числа са $\mathbb{N} = \{0, 1, 2, \dots\}$

$$\frac{P(0) \qquad (\forall x \in \mathbb{N})[P(x) \to P(x+1)]}{(\forall x \in \mathbb{N})P(x)}$$

Това означава, че ако искаме да докажем, че свойството $P(x)$ е вярно за всяко естествено число x, то трябва да докажем първо, че е изпълнено $P(0)$ и след това, за произволно естествено число x, ако $P(x)$ вярно, то също така е вярно $P(x+1)$.

Задача 1.8. Всяко естествено число $n \geq 2$ може да се запише като произведение на прости числа.

Доказателство. Искаме да докажем, че $(\forall n \geq 2)P(n)$, където $P(n)$ казва, че n може да се запише като произведение на прости числа, т.е.

$$n = p_1^{m_1} p_2^{m_2} \cdots p_k^{m_k},$$

за някои прости числа $p_1, p_2, \dots, p_k$ и естествени числа $m_1, m_2, \dots, m_k$.

Доказателството протича с индукция по $n \geq 2$.

а) За $n = 2$ е ясно, защото 2 е просто число. В този случай $n = p_1^{m_1}$ и $p_1 = 2$ и $m_1 = 1$.

б) Да приемем, че $P(n)$ е изпълнено за някое естествено число $n > 2$.

в) Да разгледаме следващото естествено число $n+1$. Ако $n+1$ е просто число, то всичко е ясно. Ако $n+1$ е съставно, то съществуват естествени числа $n_1, n_2 \geq 2$, за които

$$n+1 = n_1 \cdot n_2.$$

Тогава, понеже $n_1, n_2 \leq n$, от от И.П. следва, че $P(n_1)$ и $P(n_2)$, т.е.

$$n_1 = p_1^{\ell_1} \cdots p_k^{\ell_k} \text{ и } n_2 = q_1^{m_1} \cdots q_r^{m_r},$$

където $p_1, \ldots, p_k$ и $q_1, \ldots, q_r$ са прости числа, а $\ell_1, \ldots, \ell_k$ и $m_1, \ldots, m_r$ са естествени числа. Тогава е ясно, че $n+1$ също е произведение на прости числа.

□

Задача 1.9. Докажете, че за всяко естествено число n,

$$\sum_{i=0}^{n} 2^i = 2^{n+1} - 1.$$

Доказателство. Доказателството протича с индукция по n.

- За $n = 0$, $\sum_{i=0}^{0} 2^i = 1 = 2^1 - 1$.
- Нека твърдението е вярно за n. Ще докажем, че твърдението е вярно за $n+1$.

$$\begin{aligned}
\sum_{i=0}^{n+1} 2^i &= \sum_{i=0}^{n} 2^i + 2^{n+1} \\
&= 2^{n+1} - 1 + 2^{n+1} && \text{// от И.П.} \\
&= 2.2^{n+1} - 1 \\
&= 2^{(n+1)+1} - 1.
\end{aligned}$$

□

1.5 Азбуки, думи, езици

> Language is an app for converting a web of thoughts into a string of words. [Steven Pinker]

Основни понятия

- **Азбука** ще наричаме всяко крайно множество, като обикновено ще я означаваме със Σ. Елементите на азбуката Σ ще наричаме **букви**.

Често ще използваме буквите a, b, c за да означаваме букви.

- **Дума** над азбуката Σ е произволна крайна редица от елементи на Σ. Например, за $\Sigma = \{a, b\}$, $aababba$ е дума над Σ с дължина 7. С $|\alpha|$ ще означаваме дължината на думата α.

Обикновено ще означаваме думите с $\alpha, \beta, \gamma, \omega$.

- Обърнете внимание, че имаме единствена дума с дължина 0. Тази дума ще означаваме с ε и ще я наричаме **празната дума**, т.е. $|\varepsilon| = 0$.

- С a^n ще означаваме думата съставена от n a-та. Формалната индуктивна дефиниция е следната:

$$a^0 \stackrel{\text{деф}}{=} \varepsilon,$$
$$a^{n+1} \stackrel{\text{деф}}{=} a^n a.$$

- Множеството от всички думи над азбуката Σ ще означаваме със $\Sigma^\star$. Например, за $\Sigma = \{a, b\}$,
$$\Sigma^\star = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, \dots\}.$$
Обърнете внимание, че $\emptyset^\star = \{\varepsilon\}$.

- **Език** над азбуката Σ ще наричаме всяко подмножество на $\Sigma^\star$. Например, за азбуката $\Sigma = \{a, b\}$, множоеството от думи $L = \{\alpha \in \{a, b\}^\star \mid \alpha \text{ започва с } a\}$ е пример за език над Σ.

Операции върху думи

- Операцията **конкатенация** взима две думи α и β и образува новата дума $\alpha \cdot \beta$ като слепва двете думи. Например $aba \cdot bb = ababb$. Обърнете внимание, че в общия случай $\alpha \cdot \beta \neq \beta \cdot \alpha$. Можем да дадем формална индуктивна дефиниция на операцията конкатенация по дължината на думата β.

Често ще пишем $\alpha\beta$ вместо $\alpha \cdot \beta$

  - Ако $|\beta| = 0$, то $\beta = \varepsilon$. Тогава $\alpha \cdot \varepsilon \stackrel{\text{деф}}{=} \alpha$.
  - Ако $|\beta| = n + 1$, то $\beta = \gamma b$, $|\gamma| = n$. Тогава $\alpha \cdot \beta \stackrel{\text{деф}}{=} (\alpha \cdot \gamma)b$.

- Друга често срещана операция върху думи е **обръщането** на дума. Дефинираме думата $\alpha^{\mathtt{rev}}$ като обръщането на α по следния начин.

  Например, $reverse^{\mathtt{rev}} = esrever$

  - Ако $|\alpha| = 0$, то $\alpha = \varepsilon$ и $\alpha^{\mathtt{rev}} \stackrel{\text{деф}}{=} \varepsilon$.
  - Ако $|\alpha| = n + 1$, то $\alpha = a\beta$, където $|\beta| = n$. Тогава $\alpha^{\mathtt{rev}} \stackrel{\text{деф}}{=} (\beta^{\mathtt{rev}})a$.

- Дефинираме конкатенацията на езиците A и B като

  Обърнете внимание, че: $\emptyset \cdot A = A \cdot \emptyset = \emptyset$, $\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A$.

$$A \cdot B \stackrel{\text{деф}}{=} \{\alpha \cdot \beta \mid \alpha \in A \,\&\, \beta \in B\}.$$

- Сега за един език A, дефинираме A^n индуктивно:

$$A^0 \stackrel{\text{деф}}{=} \{\varepsilon\},$$
$$A^{n+1} \stackrel{\text{деф}}{=} A^n \cdot A.$$

  - Ако $A = \{ab, ba\}$, то $A^0 = \{\varepsilon\}$, $A^1 = A$, $A^2 = \{abab, abba, baba, baab\}$.
  - Ако $A = \{a, b\}$, то $A^n = \{\alpha \in \{a, b\}^\star \mid |\alpha| = n\}$.

- За един език A, дефинираме:

  Операцията $\star$ е известна като звезда на Клини.

$$A^\star \stackrel{\text{деф}}{=} \bigcup_{n \geq 0} A^n$$
$$= A^0 \cup A^1 \cup A^2 \cup A^3 \cup \ldots$$
$$A^+ \stackrel{\text{деф}}{=} A \cdot A^\star.$$

Пример 1.3. Нека да разгледаме няколко примера какво точно представлява прилагането на операцията звезда на Клини върху един език.

- Нека $L = \{0, 11\}$. Тогава:
  - $L^0 = \{\varepsilon\}$, $L^1 = L$,
  - $L^2 = L^1 \cdot L^1 = \{00, 011, 110, 1111\}$,
  - $L^3 = L^1 \cdot L^2 = \{000, 0011, 0110, 01111, 1100, 11011, 11110, 111111\}$.
- Нека $L = \emptyset$. Тогава $L^0 = \{\varepsilon\}$, $L^1 = \emptyset$ и $L^2 = L^1 \cdot L^1 = \emptyset$. Получаваме, че $L^\star = \{\varepsilon\}$, т.е. *краен* език
- Нека $L = \{0^i \mid i \in \mathbb{N}\} = \{\varepsilon, 0, 00, 000, \ldots\}$. Лесно се вижда, че $L = L^\star$.

Задача 1.10. Проверете:

а) операцията конкатенация е *асоциативна*, т.е. за всеки три думи α, β, γ,

$$(\alpha \cdot \beta) \cdot \gamma = \alpha \cdot (\beta \cdot \gamma);$$

б) за произволни езици A, B и C,

$$(A \cdot B) \cdot C = A \cdot (B \cdot C);$$

в) $\{\varepsilon\}^\star = \{\varepsilon\}$;

г) за произволен език A,
$$A^\star = A^\star \cdot A^\star \text{ и } (A^\star)^\star = A^\star;$$

д) за произволни букви a и b,
$$\{a,b\}^\star = \{a\}^\star \cdot (\{b\} \cdot \{a\}^\star)^\star;$$

е) за произволни букви a и b,
$$\{a,b\}^\star = (\{a\}^\star \cdot \{b\}^\star)^\star;$$

Задача 1.11. Докажете, че за всеки две думи α и β е изпълнено:

а) $(\alpha \cdot \beta)^{\mathtt{rev}} = \beta^{\mathtt{rev}} \cdot \alpha^{\mathtt{rev}}$;

б) α е префикс на β точно тогава, когато $\alpha^{\mathtt{rev}}$ е суфикс на $\beta^{\mathtt{rev}}$;

в) $(\alpha^{\mathtt{rev}})^{\mathtt{rev}} = \alpha$;

г) $(\alpha^n)^{\mathtt{rev}} = (\alpha^{\mathtt{rev}})^n$, за всяко $n \geq 0$.

Отрези на дума

Понякога може да е удобно да вземем назаем от `python` нотацията за `slices` на масив и ако думата $\alpha = a_0 a_1 \cdots a_{n-1}$, то нека

$$\begin{aligned}
\alpha\mathtt{[}i\mathtt{]} &\stackrel{\text{деф}}{=} a_i \\
\alpha\mathtt{[}i : j\mathtt{]} &\stackrel{\text{деф}}{=} \begin{cases} a_i \cdots a_{m-1}, & \text{ако } i < j \;\&\; m = \min\{j, |\alpha|\} \\ \varepsilon, & \text{иначе} \end{cases} \\
\alpha\mathtt{[}i :\mathtt{]} &\stackrel{\text{деф}}{=} \alpha\mathtt{[}i : |\alpha|\mathtt{]} \\
\alpha\mathtt{[:}\, i\mathtt{]} &\stackrel{\text{деф}}{=} \alpha\mathtt{[}0 : i\mathtt{]}
\end{aligned}$$

Релации между думи

- Казваме, че думата α е **префикс** на думата β, ако съществува дума γ, такава че $\beta = \alpha \cdot \gamma$. Да обърнем внимание, че позволяваме $\gamma = \varepsilon$. Тогава β е префикс на самата себе си. Ако се ограничим до думи $\gamma \neq \varepsilon$, то ще казваме, че α е *същински* префикс на β.

- За един език L, можем да дефинираме езика от префиксите на думите от L, т.е.
$$\mathtt{Pref}(L) = \{\alpha \in \Sigma^\star \mid (\exists \gamma \in \Sigma^\star)[\alpha \cdot \gamma \in L]\}.$$

- α е **суфикс** на β, ако $\beta = \gamma \cdot \alpha$, за някоя дума γ.

- За един език L, можем да дефинираме езика от суфиксите на думите от L, т.е.
$$\mathtt{Suff}(L) = \{\alpha \in \Sigma^\star \mid (\exists \gamma \in \Sigma^\star)[\gamma \cdot \alpha \in L]\}.$$

- Нека имаме азбука $\Sigma = \{0, 1, 2, \ldots, n\}$. Казваме, че една дума α е лексикографски по-малка от β, което ще означаваме като $\alpha <_{\mathtt{lex}} \beta$, ако α е префикс на β или
$$(\exists i < \min\{|\alpha|, |\beta|\})[\, \alpha\mathtt{[:}\, i\mathtt{]} = \beta\mathtt{[:}\, i\mathtt{]} \;\&\; \alpha\mathtt{[}i\mathtt{]} < \beta\mathtt{[}i\mathtt{]}\,].$$

Тук не е съществено, че буквите в азбуката Σ са числа. Можем да дефинираме наредба между елементите на произволна крайна азбука.

1.6 Дървета

Понятието дърво е едно от най-основните в информатиката и може да се дефинира по много различни начини, в зависимост от това за какви цели се използва. Тук на практика следваме [NS93], защото по-късно ще е важно да имаме наредба между възлите на дървото. При нас всички дървета са крайно разклонени и всеки възел в дървото еднозначно се определя от пътя от възела до корена. Обърнете внимание, че тук дърветата, макар и крайно разклонени, могат да бъдат безкрайни.

Нека $b \in \mathbb{N}$. Непразното множество $T \subseteq \{0, 1, \ldots, b-1\}^\star$ се нарича **дърво**, ако T е затворено относно префикси, т.е. $\texttt{Pref}(T) = T$. С други думи,

$$(\forall \alpha)(\forall \beta)[\, \alpha \in T \,\&\, \beta \preceq \alpha \implies \beta \in T \,].$$

Ако гледаме на елементите на T като върхове на граф, то можем да дефинираме бинарната релацията $Succ$, върху едно дърво T като

$$Succ(\alpha, \beta) \overset{\text{деф}}{\Leftrightarrow} \alpha, \beta \in T \,\&\, (\exists c < b)[\alpha \cdot c = \beta],$$

която казва, че върха β е пряк наследник на върха α. Обърнете внимание, че е възможно $T \neq T'$ да са такива, че $(T, Succ) \cong (T', Succ')$.

Поради тази причина е удобно да наречем едно дърво T *оптимално*, ако за него е изпълнено свойството:

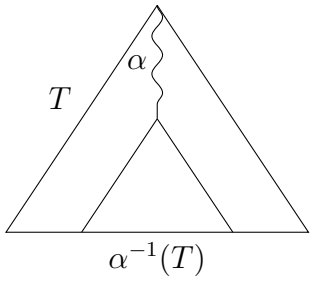
$$\alpha \cdot c \in T \,\&\, a < c \implies \alpha \cdot a \in T. \tag{1.1}$$

Тогава лесно се вижда, че за две оптимални дървета T и T',

$$T = T' \Leftrightarrow (T, Succ) \cong (T', Succ').$$

Нека да въведем следните означения:

$$\begin{aligned}
&\texttt{height}(T) \overset{\text{деф}}{=} \max\{\, |\alpha| \mid \alpha \in T \,\} && \text{// височина} \\
&\texttt{succ}_T(\alpha) \overset{\text{деф}}{=} \{\alpha c \mid \alpha c \in T \,\&\, c < b\} && \text{// наследниците на } \alpha \\
&T_\alpha \overset{\text{деф}}{=} \alpha^{-1}(T) && \text{// поддървото на } \alpha \\
&\texttt{leaves}(T) \overset{\text{деф}}{=} \{\alpha \in T \mid \texttt{succ}_T(\alpha) = \emptyset\}. && \text{// листата на } T
\end{aligned}$$



Фигура 1.1: $\alpha^{-1}(T)$ е поддърво на T.

Задача 1.12. Докажете, че ако T е дърво, то $\texttt{Pref}(\texttt{leaves}(T)) = T$.

Задача 1.13. Нека $T \subseteq \{0, \ldots, b-1\}^\star$ е крайно дърво. Докажете, че

$$|\texttt{leaves}(T)| \leq b^{\texttt{height}(T)}.$$

Доказателство. Индукция по $\texttt{height}(T)$.

- Нека $\texttt{height}(T) = 0$. Тогава е ясно, че $|\texttt{leaves}(T)| = |\{\varepsilon\}| = 1 \leq b^0$.
- Нека $\texttt{height}(T) > 0$. За всяко $a \in T$ е ясно, че $\texttt{height}(T_a) < \texttt{height}(T)$. Тогава:

В този случай имаме, че $\texttt{leaves}(T) = \bigcup_{a \in T}(\{a\} \cdot \texttt{leaves}(T_a))$. Тук гледаме на a и b от една страна като букви в азбука, но и като числа.

$$\begin{aligned}
|\texttt{leaves}(T)| &= \sum_{a \in T} |\texttt{leaves}(T_a)| && \text{// } T_a \stackrel{\text{деф}}{=} a^{-1}(T) \\
&\leq \sum_{a \in T} b^{\texttt{height}(T_a)} && \text{// от } \textbf{(И.П.)} \\
&\leq \sum_{a \in T} b^{\texttt{height}(T)-1} && \text{// } \texttt{height}(T_a) \leq \texttt{height}(T) - 1 \\
&\leq \sum_{a < b} b^{\texttt{height}(T)-1} \\
&= b^{\texttt{height}(T)}.
\end{aligned}$$

□

$f \restriction n \stackrel{\text{деф}}{=} f(0)f(1)\cdots f(n-1)$.

Твърдение 1.1 (Лема на Кьониг)**.** Нека $T \subseteq \{0, 1, \ldots, k\}^\star$. Ако T е безкрайно дърво, то T съдържа безкраен път, т.е. съществува $\pi : \mathbb{N} \to \{0, 1, \ldots, k\}$, такава че $f \restriction n \in T$ за всяко $n \in \mathbb{N}$.

Упътване. Дефинираме безкрайния път π на стъпки. На всяка стъпка избираме този наследник, който е корен на безкрайно дърво. Понеже T е безкрайно дърво с крайно разклонение, на всяка стъпка можем да изберем такъв наследник. □

Доказателство. По-формално, ще дефинираме редица от думи $\alpha_0 \prec \alpha_1 \prec \cdots \prec \alpha_n \prec \cdots$, които ще образуват безкрайния път, т.е. $f \restriction n = \alpha_n$, като искаме, за всяко n, $T_n \stackrel{\text{деф}}{=} (\alpha_n)^{-1}(T)$ да бъде безкрайно дърво. Нека $\alpha_0 = \varepsilon$, коренът на T. Ясно е, че по условие, $T_0 = T$ е безкрайно дърво. Да приемем, че сме дефинирали $\alpha_0 \prec \alpha_1 \prec \cdots \prec \alpha_n$ и T_n е безкрайно дърво. Ще дефинираме α_{n+1}. Понеже T_n е крайно разклонено дърво с максимално разклонение $k+1$, възелът α_n в дървото има крайно много наследници. Понеже T_n е безкрайно дърво, то поне един от наследниците, да кажем $y \leq k$, на α_n ще има безкрайно много наследници. Нека $\alpha_{n+1} \stackrel{\text{деф}}{=} \alpha_n \cdot y$. Ясно е, че T_{n+1} е безкрайно дърво. □

Глава 2

Регулярни езици и автомати

2.1 Автоматни езици

Определение 2.1. Детерминиран краен автомат е петорка $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$, където

Често ще пишем съкратено ДКА вместо детерминиран краен автомат.

- Σ е азбука;
- Q е крайно множество от състояния;
- $\delta : Q \times \Sigma \to Q$ е тотална функция, която ще наричаме *функция на преходите*;
- $q_{\texttt{start}} \in Q$ е начално състояние;
- $F \subseteq Q$ е множеството от финални състояния

Нека имаме една дума $\alpha \in \Sigma^\star$, $\alpha = a_0 a_1 \cdots a_{n-1}$. Казваме, че α се **разпознава** от автомата $\mathcal{A}$, ако съществува редица от състояния $q_0, q_1, q_2, \ldots, q_n$, такива че:

- $q_0 = q_{\texttt{start}}$, началното състояние на автомата;
- $\delta(q_i, a_i) = q_{i+1}$, за всяко $i = 0, \ldots, n-1$;
- $q_n \in F$.

Казваме, че $\mathcal{A}$ **разпознава** езика L, ако $\mathcal{A}$ разпознава точно думите от L, т.е. $L = \{\alpha \in \Sigma^\star \mid \mathcal{A} \text{ разпознава } \alpha\}$. Обикновено означаваме езика, който се разпознава от даден автомат $\mathcal{A}$ с $\mathcal{L}(\mathcal{A})$. В такъв случай ще казваме, че езикът L е **автоматен**.

При дадена функция на преходите δ, често е удобно да разглеждаме функция $\delta^\star : Q \times \Sigma^\star \to Q$, която е дефинирана за всяко $q \in Q$ и $\alpha \in \Sigma^\star$ по следния начин:

Дали не е по-добре да се дефинира като $\delta^\star(q, \varepsilon) = q$ и $\delta^\star(q, a\beta) = \delta^\star(\delta(q, a), \beta)$?

- Ако $\alpha = \varepsilon$, то $\delta^\star(q, \varepsilon) \stackrel{\text{деф}}{=} q$;
- Ако $\alpha = \beta a$, то $\delta^\star(q, \beta a) \stackrel{\text{деф}}{=} \delta(\delta^\star(q, \beta), a)$.

Лесно се съобразява, че една дума α се *разпознава* от автомата $\mathcal{A}$ точно тогава, когато $\delta^\star(q_{\texttt{start}}, \alpha) \in F$. Оттук следва, че

$$\mathcal{L}(\mathcal{A}) \stackrel{\text{деф}}{=} \{\, \alpha \in \Sigma^\star \mid \delta^\star(q_{\texttt{start}}, \alpha) \in F \,\}.$$

Обърнете внимание, че $\delta(q, a) = \delta^\star(q, a)$ за $a \in \Sigma$

Твърдение 2.1. Нека $\mathcal{A}$ е ДКА. Тогава за всяко състояние q и произволни думи α и β е изпълнено, че

$$\delta^\star(q, \alpha\beta) = \delta^\star(\delta^\star(q, \alpha), \beta).$$

Упътване. Индукция по дължината на думата β.

- $|\beta| = 0$, т.е. $\beta = \varepsilon$. Тогава за всяко състояние q и произволна дума α имаме:

$$\delta^\star(q, \alpha\varepsilon) = \delta^\star(\underbrace{\delta^\star(q, \alpha)}_{q}, \varepsilon).$$

- Да приемем, че твърдението е изпълнено за думи β с дължина n.

- Нека $|\beta| = n + 1$, т.е. $\beta = \gamma b$, където $|\gamma| = n$. Тогава за всяко състояние q и произволна дума α имаме:

$$\begin{aligned}
\delta^\star(q, \alpha\beta) &= \delta^\star(q, \alpha\gamma b) \\
&= \delta(\delta^\star(q, \alpha\gamma), b) && \text{// от деф. на } \delta^\star \\
&= \delta(\delta^\star(\underbrace{\delta^\star(q, \alpha)}_{p}, \gamma), b) && \text{// от И.П. приложено за } \gamma \\
&= \delta(\delta^\star(p, \gamma), b) \\
&= \delta^\star(p, \gamma b) && \text{// от деф. на } \delta^\star \\
&= \delta^\star(\delta^\star(q, \alpha), \beta). && \text{// } p = \delta^\star(q, \alpha)
\end{aligned}$$

□

Забележка. Формално погледнато, доказателството на *Твърдение 2.1* протича по следната схема:

$$\frac{P(0) \qquad (\forall n \in \mathbb{N})[P(n) \implies P(n+1)]}{(\forall n \in \mathbb{N})[P(n)],}$$

където свойството P е дефинирано така:

$$P(n) \stackrel{\text{деф}}{=} (\forall \beta \in \Sigma^n)(\forall \alpha \in \Sigma^\star)(\forall q \in Q)[\delta^\star(q, \alpha\beta) = \delta^\star(\delta^\star(q, \alpha), \beta)].$$

Конфигурация (или моментното описание) представлява описание на текущото състояние на едно изчисление с краен автомат. То представлява двойка от вида $(q, \alpha) \in Q \times \Sigma^\star$, т.е. автоматът се намира в състояние q, а думата, която остава да се прочете е α. Удобно е да въведем бинарната релация $\vdash_{\mathcal{A}}$ над $Q \times \Sigma^\star$, която ще ни казва как моментното описание на автомата $\mathcal{A}$ се променя след изпълнение на една стъпка.

(На англ. *instantaneous description*). В случая въвеждането на това понятие не е напълно необходимо, но по-късно, когато въведем стекови автомати и машини на Тюринг, ще работим с такива конфигурации на изчисления и затова е добре още отначало да свикнем с това понятие.

$$\frac{\delta(q, b) = p}{(q, b\alpha) \vdash_{\mathcal{A}} (p, \alpha)}$$

Фигура 2.1: Едностъпков преход в детерминиран краен автомат $\mathcal{A}$

Удобно е също така да дефинираме бинарната релация $\vdash^{\ell}_{\mathcal{A}}$ върху $Q \times \Sigma^\star$, която ни казва, че конфигурацията κ се променя до конфигурация κ' след ℓ стъпки от изчислението на автомата $\mathcal{A}$.

$$\frac{}{\kappa \vdash^{0}_{\mathcal{A}} \kappa} \text{ (рефлексивност)} \qquad \frac{\kappa \vdash_{\mathcal{A}} \kappa'' \qquad \kappa'' \vdash^{\ell}_{\mathcal{A}} \kappa'}{\kappa \vdash^{\ell+1}_{\mathcal{A}} \kappa'} \text{ (транзитивност)}$$

Дефинираме релацията $\vdash^{\star}_{\mathcal{A}}$ като рефлексивното и транзитивно затваряне на релацията $\vdash_A$, или с други думи:

$$\kappa \vdash^{\star}_{\mathcal{A}} \kappa' \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[\kappa \vdash^{\ell}_{\mathcal{A}} \kappa'].$$

Задача 2.1. Докажете, че за произволна дума $\beta \in \Sigma^\star$,

$$(q, \alpha\beta) \vdash^{\star}_{\mathcal{A}} (p, \beta) \Leftrightarrow \delta^\star(q, \alpha) = p.$$

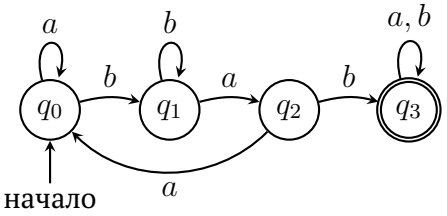
Можем да заключим, че имаме следното преставяне:

$$\mathcal{L}(\mathcal{A}) = \{\alpha \in \Sigma^\star \mid (\exists f \in F)[(q_{\texttt{start}}, \alpha) \vdash^{\star}_{\mathcal{A}} (f, \varepsilon)]\}.$$

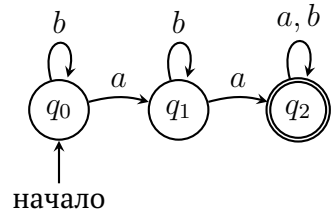
Примерни задачи

Винаги с удвоени окръжности ще отбелязваме финалните състояния. За момента нямаме общ метод, с който да докажем, че даденият автомат разпознава точно съответния език.

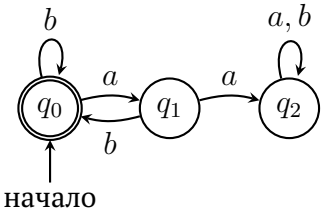
Пример 2.1. Да разгледаме няколко примера за автомати и езиците, които разпознават. Дефинирайте функцията на преходите δ за всеки автомат.



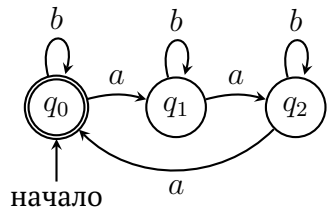
(а) $\{\omega \in \{a,b\}^\star \mid \omega \text{ съдържа } bab\}$



(б) $\{\omega \in \{a,b\}^\star \mid |\omega|_a \geq 2\}$



(в) $\{\omega \in \{a,b\}^\star \mid$ всяко a в ω веднага се следва от поне едно $b\}$



(г) $\{\omega \in \{a,b\}^\star \mid |\omega|_a \equiv 0 \mod 3\}$

В повечето от горните примери може сравнително лесно да се съобрази, че построеният автомат разпознава желания език. При по-сложни задачи обаче, ще се наложи да дадем доказателство, като обикновено се прилага *метода на математическата индукция* върху дължината на думите. Ще разгледаме няколко такива примера.

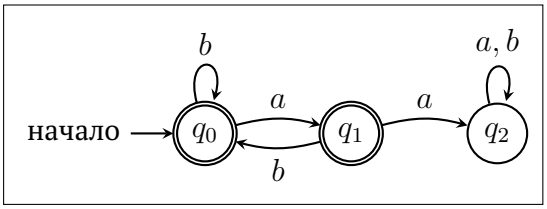
Най-лесния начин да се сетим как да построим $\mathcal{A}$ е като първо построим автомат за езика, който разпознава тези думи, в които се съдържат две поредни срещания на a и вземем неговото допълнение съгласно Твърдение 2.3. По-късно ще разгледаме общ метод за строене на автомат по език.

Задача 2.2. Докажете, че езикът

$$L = \{\alpha \in \{a,b\}^\star \mid \alpha \text{ не съдържа две поредни срещания на } a\}$$

е автоматен.

Доказателство. Да разгледаме $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ с функция на преходите описана на *Фигура 2.4*.



Фигура 2.4: Автомат $\mathcal{A}$ разпознаващ думите, които не съдържат две поредни срещания на a

$|\alpha| \stackrel{\text{деф}}{=}$ дължината на α.

Ще докажем, че $L = \mathcal{L}(\mathcal{A})$. Първо ще се концентрираме върху доказателството на $\mathcal{L}(\mathcal{A}) \subseteq L$. Ще докажем, че за всяка дума $\alpha \in \Sigma^\star$ са изпълнени свойствата:

(1) ако $\delta^\star(q_0, \alpha) = q_0$, то α не съдържа две поредни срещания на a и ако $|\alpha| > 0$, то α завършва на b;

(2) ако $\delta^\star(q_0, \alpha) = q_1$, то α не съдържа две поредни срещания на a и завършва на a.

Ще докажем (1) и (2) едновременно с индукция по дължината на думата α.

- За $|\alpha| = 0$, т.е. $\alpha = \varepsilon$, твърденията (1) и (2) са ясни (Защо?).
- Да приемем, че твърденията (1) и (2) са верни за произволни думи α с дължина n.
- Нека $|\alpha| = n + 1$, т.е. $\alpha = \beta x$, където $|\beta| = n$ и $x \in \Sigma$. Ще докажем (1) и (2) за α.
  - Нека $\delta^\star(q_0, \beta x) = q_0 = \delta(\delta^\star(q_0, \beta), x)$. Според дефиницията на функцията δ, $x = b$ и $\delta^\star(q_0, \beta) \in \{q_0, q_1\}$. Тогава по **И.П.** за (1) и (2), β не съдържа две поредни срещания на a. Тогава е очевидно, че βx също не съдържа две поредни срещания на a.
  - Нека $\delta^\star(q_0, \beta x) = q_1 = \delta(\delta^\star(q_0, \beta), x)$. Според дефиницията на δ, $x = a$ и $\delta^\star(q_0, \beta) = q_0$. Тогава по **И.П.** за (1), β не съдържа две поредни срещания на a и ако $|\beta| > 0$, то β завършва на b. Тогава е очевидно, че βx също не съдържа две поредни срещания на a.

Така доказахме с индукция по дължината на думата, че за всяка дума α са изпълнени твърденията (1) и (2). По дефиниция, ако $\alpha \in \mathcal{L}(\mathcal{A})$, то $\delta^\star(q_0, \alpha) \in F = \{q_0, q_1\}$ и от (1) и (2) следва, че и в двата случа α не съдържа две поредни срещания на буквата a, т.е. $\alpha \in L$. С други думи, доказахме, че

$$\mathcal{L}(\mathcal{A}) \subseteq L.$$

Сега ще докажем другата посока, т.е. $L \subseteq \mathcal{L}(\mathcal{A})$. Това означава да докажем, че

$$(\forall \alpha \in \Sigma^\star)[\alpha \in L \;\Rightarrow\; \delta^\star(q_0, \alpha) \in F],$$

което е еквивалентно на

Да напомним, че от съждителното смятане знаем, че $p \to q \equiv \neg q \to \neg p$.

$$(\forall \alpha \in \Sigma^\star)[\delta^\star(q_0, \alpha) \notin F \;\Rightarrow\; \alpha \notin L]. \tag{2.1}$$

Това е лесно да се съобрази. Щом $\delta^\star(q_0, \alpha) \notin F$, то $\delta^\star(q_0, \alpha) = q_2$ и думата α може да се представи по следния начин:

$$\alpha = \beta a \gamma \;\&\; \delta^\star(q_0, \beta) = q_1.$$

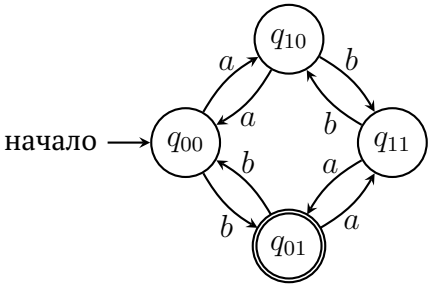
Използвайки свойство (2) от по-горе, понеже $\delta^\star(q_0, \beta) = q_1$, то β не съдържа две поредни срещания на a, но завършва на a. Сега е очевидно, че βa съдържа две поредни срещания на a и щом βa е префикс на α, то думата $\alpha \notin L$. С това доказахме Свойство 2.1, а следователно и посоката $L \subseteq \mathcal{L}(\mathcal{A})$. □

Задача 2.3. Докажете, че езикът

$$L = \{\omega \in \{a,b\}^\star \mid a \text{ се среща четен брой, докато } b \text{ нечетен брой пъти в } \omega\}$$

е автоматен.

Упътване. Разгледайте автомата $\mathcal{A}$:



Фигура 2.5: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} L$

$|\omega|_a \stackrel{\text{деф}}{=}$ броят на срещанията на буквата a в думата ω.

Докажете с индукция по дължината на думата ω, че:

а) $\delta^\star(q_{00},\omega) = q_{00} \implies |\omega|_a \equiv 0 \bmod 2 \;\&\; |\omega|_b \equiv 0 \bmod 2$;

б) $\delta^\star(q_{00},\omega) = q_{01} \implies |\omega|_a \equiv 0 \bmod 2 \;\&\; |\omega|_b \equiv 1 \bmod 2$;

в) $\delta^\star(q_{00},\omega) = q_{10} \implies |\omega|_a \equiv 1 \bmod 2 \;\&\; |\omega|_b \equiv 0 \bmod 2$;

г) $\delta^\star(q_{00},\omega) = q_{11} \implies |\omega|_a \equiv 1 \bmod 2 \;\&\; |\omega|_b \equiv 1 \bmod 2$;

Оттук направете извода, че за произволна дума ω,

$$(\forall i < 2)(\forall j < 2)[\, \delta^\star(q_{00},\omega) = q_{ij} \;\Leftrightarrow\; |\omega|_a \equiv i \bmod 2 \;\&\; |\omega|_b \equiv j \bmod 2\,].$$

□

За една дума $\alpha \in \{0,1\}^\star$, нека с $\overline{\alpha}_{(k)}$ да означим числото, което се представя в k-ична бройна система като α. Например,

$$\overline{1101}_{(2)} = 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = \overline{13}_{(10)} = 1.10^1 + 3.10^0.$$

Да отбележим, че за всяко число n има безкрайно много думи α, за които $\overline{\alpha}_{(2)} = n$. Например,

$\overline{10}_{(2)} = \overline{010}_{(2)} = \overline{0010}_{(2)} = \cdots$

За $k = 2$, имаме следната дефиниция:

- $\overline{\varepsilon}_{(2)} = 0$,
- $\overline{\alpha 0}_{(2)} = 2 \cdot \overline{\alpha}_{(2)}$,
- $\overline{\alpha 1}_{(2)} = 2 \cdot \overline{\alpha}_{(2)} + 1$.

Задача 2.4. Докажете, че езикът $L = \{\alpha \in \{0,1\}^\star \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\}$ е автоматен.

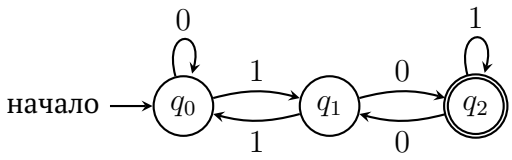
Доказателство. Нашият автомат ще има три състояния $\{q_0, q_1, q_2\}$, като началното състояние ще бъде q_0. Целта ни е да дефинираме така автомата, че да имаме следното свойство:

$$(\forall \alpha \in \Sigma^\star)(\forall i < 3)[\, \overline{\alpha}_{(2)} \equiv i \bmod 3 \;\Leftrightarrow\; \delta^\star(q_0, \alpha) = q_i \,], \tag{2.2}$$

т.е. всяко състояние отговаря на определен остатък при деление на три. Понеже искаме нашия автомат да разпознава тези думи α, за които $\alpha_{(2)} \equiv 2 \mod 3$, финалното състояние ще бъде q_2. Дефинираме функцията δ по следния начин:

$$\begin{aligned}
&\delta(q_0, 0) = q_0 && // \ \overline{\alpha}_{(2)} \equiv 0 \bmod 3 \implies \overline{\alpha 0}_{(2)} \equiv 0 \bmod 3\\
&\delta(q_0, 1) = q_1 && // \ \overline{\alpha}_{(2)} \equiv 0 \bmod 3 \implies \overline{\alpha 1}_{(2)} \equiv 1 \bmod 3\\
&\delta(q_1, 0) = q_2 && // \ \overline{\alpha}_{(2)} \equiv 1 \bmod 3 \implies \overline{\alpha 0}_{(2)} \equiv 2 \bmod 3\\
&\delta(q_1, 1) = q_0 && // \ \overline{\alpha}_{(2)} \equiv 1 \bmod 3 \implies \overline{\alpha 1}_{(2)} \equiv 0 \bmod 3\\
&\delta(q_2, 0) = q_1 && // \ \overline{\alpha}_{(2)} \equiv 2 \bmod 3 \implies \overline{\alpha 0}_{(2)} \equiv 1 \bmod 3\\
&\delta(q_2, 1) = q_2 && // \ \overline{\alpha}_{(2)} \equiv 2 \bmod 3 \implies \overline{\alpha 1}_{(2)} \equiv 2 \bmod 3
\end{aligned}$$

Ето и картинка на автомата $\mathcal{A}$:



Фигура 2.6: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \{\alpha \in \{0,1\}^\star \mid \overline{\alpha}_{(2)} \equiv 2 \,(\mathrm{mod}\, 3)\}$

Да разгледаме твърденията:

(1) $\delta^\star(q_0, \alpha) = q_0 \implies \overline{\alpha}_{(2)} \equiv 0 \mod 3$;

(2) $\delta^\star(q_0, \alpha) = q_1 \implies \overline{\alpha}_{(2)} \equiv 1 \mod 3$;

(3) $\delta^\star(q_0, \alpha) = q_2 \implies \overline{\alpha}_{(2)} \equiv 2 \mod 3$.

Ще докажем (1), (2) и (3) *едновременно* с индукция по дължината на думата α. За $|\alpha| = 0$, всички условия са изпълнени. (Защо?) Да приемем, че (1), (2) и (3) са изпълнени за думи с дължина n. Нека $|\alpha| = n + 1$, т.е. $\alpha = \beta x$, $|\beta| = n$.

Обърнете внимание, че в доказателството на (3) използваме И.П. не само за (3), но и за (2). Поради тази причина трябва да докажем трите свойства едновременно

Ще докажем подробно само (3) понеже другите твърдения се доказват по сходен начин. Нека $\delta^\star(q_0, \beta x) = q_2$. Имаме два случая:

- $x = 0$. Тогава, по дефиницията на δ, $\delta(q_1, 0) = q_2$ и следователно, $\delta^\star(q_0, \beta) = q_1$. По **И.П.** за (2) с β имаме импликацията:

  $$\delta^\star(q_0, \beta) = q_1 \;\Rightarrow\; \overline{\beta}_{(2)} \equiv 1 \bmod 3$$

  Тогава, $\overline{\beta 0}_{(2)} \equiv 2 \mod 3$. Така доказахме, че

  $$\delta^\star(q_0, \beta 0) = q_2 \;\Rightarrow\; \overline{\beta 0}_{(2)} \equiv 2 \bmod 3.$$

- $x = 1$. Тогава, по дефиницията на δ, $\delta(q_2, 1) = q_2$ и следователно, $\delta^\star(q_0, \beta) = q_2$. По **И.П.** за (3) с β имаме импликацията:

$$\delta^\star(q_0, \beta) = q_2 \;\Rightarrow\; \overline{\beta}_{(2)} \equiv 2 \bmod 3.$$

Тогава, $\overline{\beta 1}_{(2)} \equiv 2 \mod 3$. Така доказахме, че

$$\delta^\star(q_0, \beta 1) = q_2 \;\Rightarrow\; \overline{\beta 1}_{(2)} \equiv 2 \bmod 3.$$

✍ Довършете доказателствата на (1) и (2)

За да докажем (1), нека $\delta^\star(q_0, \beta x) = q_0$.

- $x = 0$. Разсъжденията са аналогични, като използваме **И.П.** за (1).
- $x = 1$. Разсъжденията са аналогични, като използваме **И.П.** за (2).

По същия начин доказваме и (2). Нека $\delta^\star(q_0, \beta x) = q_1$.

- При $x = 0$, използваме **И.П.** за (3).
- При $x = 1$, използваме **И.П.** за (1).

✍ Защо?

От (1), (2) и (3) следва директно, че е изпълнено свойството:

$$(\forall \alpha \in \Sigma^\star)(\forall i < 3)[\, \overline{\alpha}_{(2)} \equiv i \,(\mathrm{mod}\ 3) \;\Leftrightarrow\; \delta^\star(q_0, \alpha) = q_i \,],$$

откъдето получаваме, че $\mathcal{L}(\mathcal{A}) = L$. □

Затвореност относно сечение, обединение, разлика

Твърдение 2.2. Класът на автоматните езици е затворен относно операцията *сечение*. Това означава, че ако L_1 и L_2 са два произволни автоматни езика над азбуката Σ, то $L_1 \cap L_2$ също е автоматен език.

Доказателство. Да разгледаме два автомата

$$\mathcal{A}_1 = \langle \Sigma, Q_1, q'_{\mathtt{start}}, \delta_1, F_2 \rangle \text{ и } \mathcal{A}_2 = \langle \Sigma, Q_2, q''_{\mathtt{start}}, \delta_2, F_2 \rangle.$$

Изчислението на автомата $\mathcal{A}$ върху думата α едновременно симулира изчислението на $\mathcal{A}_1$ и $\mathcal{A}_2$ върху α.

Определяме автомата $\mathcal{A} = \langle \Sigma, Q, q_{\mathtt{start}}, \delta, F \rangle$, по следния начин:

- $Q \stackrel{\text{деф}}{=} Q_1 \times Q_2$;

Съобразете, че δ е тотална функция.

- За всяко $\langle r_1, r_2 \rangle \in Q$ и всяко $a \in \Sigma$,

$$\delta(\langle r_1, r_2 \rangle, a) \stackrel{\text{деф}}{=} \langle \delta_1(r_1, a), \delta_2(r_2, a) \rangle;$$

- $q_{\texttt{start}} \stackrel{\text{деф}}{=} \langle q'_{\texttt{start}}, q''_{\texttt{start}} \rangle$;
- $F \stackrel{\text{деф}}{=} \{\langle r_1, r_2\rangle \mid r_1 \in F_1 \,\&\, r_2 \in F_2\} = F_1 \times F_2$

Трябва да докажем, че за всяка дума $\alpha \in \Sigma^\star$ е изпълнено, че:

$$(\forall p \in Q_1)(\forall q \in Q_2)[\, \delta^\star(\langle p, q\rangle, \alpha) = \langle \delta_1^\star(p, \alpha), \delta_2^\star(q, \alpha)\rangle \,]. \tag{2.3}$$

Ще докажем *Свойство (2.3)* с индукция по дължината на думата α.

- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Тогава всичко е ясно, защото

$$\begin{aligned} \delta^\star(\langle p, q\rangle, \varepsilon) &= \langle p, q\rangle && \text{// деф. на } \delta^\star \\ &= \langle \delta_1^\star(p, \varepsilon), \delta_2^\star(q, \varepsilon)\rangle. && \text{// деф. на } \delta_1^\star \text{ и } \delta_2^\star \end{aligned}$$

- Да приемем, че *Свойство (2.3)* е изпълнено за думи α с дължина n.
- Нека $|\alpha| = n + 1$, т.е. $\alpha = \beta a$ и $|\beta| = n$. Тогава:

$$\begin{aligned} \delta^\star(\langle p, q\rangle, \beta a) &= \delta(\delta^\star(\langle p, q\rangle, \beta), a) && \text{// деф. на } \delta^\star \\ &= \delta(\langle \delta_1^\star(p, \beta), \delta_2^\star(q, \beta)\rangle, a) && \text{// от И.П.} \\ &= \langle \delta_1(\delta_1^\star(p, \beta), a), \delta_2(\delta_2^\star(q, \beta), a)\rangle && \text{// деф. на } \delta \\ &= \langle \delta_1^\star(p, \beta a), \delta_2^\star(q, \beta a)\rangle && \text{// деф. на } \delta_1^\star \text{ и } \delta_2^\star. \end{aligned}$$

Използвайки *Свойство (2.3)* лесно можем да докажем, че

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2).$$

Имаме следните еквивалентности:

$$\begin{aligned} \omega \in \mathcal{L}(\mathcal{A}) &\Leftrightarrow \delta^\star(q_{\texttt{start}}, \omega) \in F && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \delta^\star(\langle q'_{\texttt{start}}, q''_{\texttt{start}}\rangle, \omega) \in F_1 \times F_2 && \text{// деф. на } \mathcal{A} \\ &\Leftrightarrow \langle \delta_1^\star(q'_{\texttt{start}}, \omega), \delta_2^\star(q''_{\texttt{start}}, \omega)\rangle \in F_1 \times F_2 && \text{// от (2.3)} \\ &\Leftrightarrow \delta_1^\star(q'_{\texttt{start}}, \omega) \in F_1 \,\&\, \delta_2^\star(q''_{\texttt{start}}, \omega) \in F_2 \\ &\Leftrightarrow \omega \in \mathcal{L}(\mathcal{A}_1) \,\&\, \omega \in \mathcal{L}(\mathcal{A}_2) \\ &\Leftrightarrow \omega \in \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2). \end{aligned}$$

□

Твърдение 2.3. Класът на автоматните езици са затворени относно операцията *допълнение*, т.е. ако L е автоматен език, то $\Sigma^\star \setminus L$ също е автоматен език.

✍ Проверете, че $\Sigma^\star \setminus L = \mathcal{L}(\mathcal{A}')$. Съобразете, че тук е важно, че δ е тотална функция на преходите, а не просто частична функция.

Упътване. Нека $L = L(\mathcal{A})$, където $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$. Да вземем автомата

$$\mathcal{A}' = \langle Q, \Sigma, q_{\texttt{start}}, \delta, Q \setminus F \rangle,$$

т.е. $\mathcal{A}'$ е същия като $\mathcal{A}$, с единствената разлика, че финалните състояния на $\mathcal{A}'$ са тези състояния, които **не** са финални в $\mathcal{A}$. □

Твърдение 2.4. Класът на автоматните езици е затворен относно операцията *обединение*. Това означава, че ако L_1 и L_2 са два произволни автоматни езика, то $L_1 \cup L_2$ също е автоматен език.

✍ Докажете, че така построения автомат $\mathcal{A}$ разпознава $L_1 \cup L_2$. Тук отново е важно, че δ_1 и δ_2 са тотални функции на преходите.

Упътване. Първият подход е да използвайте конструкцията на автомата $\mathcal{A}$ от *Твърдение 2.2*, с единствената разлика, че тук избираме финалните състояния да бъдат елементите на множеството

$$\begin{aligned} F &\stackrel{\text{деф}}{=} \{\langle q_1, q_2 \rangle \in Q_1 \times Q_2 \mid q_1 \in F_1 \ \vee \ q_2 \in F_2\} \\ &= F_1 \times Q_2 \cup Q_1 \times F_2. \end{aligned}$$

Друг подход е да се използва правилото на Де Морган, а именно:

$$L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}.$$

□

2.2 Регулярни изрази и езици

Да фиксираме една непразна азбука Σ. **Регулярните изрази** $\mathbf{r}$ могат да се опишат със следната абстрактна граматика

$$\mathbf{r} ::= \boldsymbol{\emptyset} \mid \boldsymbol{\varepsilon} \mid \mathbf{a} \mid (\boldsymbol{r}_1 \cdot \boldsymbol{r}_2) \mid (\boldsymbol{r}_1 + \boldsymbol{r_2}) \mid \boldsymbol{r}_1^\star.$$

Регулярните изрази могат да се опишат и по следния начин: Това е пример за индуктивна дефиниция

- Символите $\boldsymbol{\emptyset}$, $\boldsymbol{\varepsilon}$ са регулярни изрази;
- за всяка буква $a \in \Sigma$, символът $\boldsymbol{a}$ е регулярен израз;
- ако $\mathbf{r_1}$ и $\mathbf{r_2}$ са регулярни изрази, то думите $(\boldsymbol{r}_1 \cdot \boldsymbol{r}_2)$, $(\boldsymbol{r}_1 + \boldsymbol{r}_2)$ и $\boldsymbol{r}_1^\star$ също са регулярни изрази; В литературата също се среща записът $(\boldsymbol{r}_1 \mid \boldsymbol{r}_2)$ вместо $(\boldsymbol{r}_1 + \boldsymbol{r}_2)$
- Всеки регулярен израз е получен по някое от горните правила.

Сега ще дефинираме езиците, които се описват с регулярни изрази. Тези езици се наричат **регулярни**. Това ще направим следвайки индуктивната дефиниция на регулярните изрази, т.е. за всеки регулярен израз $\mathbf{r}$ ще определим език $\mathcal{L}(\mathbf{r})$. Това е друг пример за индуктивна (рекурсивна) дефиниция.

- $\emptyset$ е регулярен език, който се описва от регулярния израз $\boldsymbol{\emptyset}$. Означаваме
$$\mathcal{L}(\boldsymbol{\emptyset}) \stackrel{\text{деф}}{=} \emptyset;$$
- $\{\varepsilon\}$ е регулярен език, който се описва от регулярния израз $\boldsymbol{\varepsilon}$. Означаваме
$$\mathcal{L}(\boldsymbol{\varepsilon}) \stackrel{\text{деф}}{=} \{\varepsilon\};$$
- за всяка буква $a \in \Sigma$, $\{a\}$ е регулярен език, който се описва от регулярния израз $\mathbf{a}$. Означаваме
$$\mathcal{L}(\mathbf{a}) \stackrel{\text{деф}}{=} \{a\};$$
- Нека L_1 и L_2 са регулярни езици, т.е. съществуват регулярни изрази $\mathbf{r}_1$ и $\mathbf{r}_2$, за които $\mathcal{L}(\mathbf{r}_1) = L_1$ и $\mathcal{L}(\mathbf{r}_2) = L_2$. Тогава: Понякога, когато приоритетът на операциите е ясен, ще изпускаме да пишем скоби.
  - $L_1 \cup L_2$ е регулярен език, който се описва с регулярния израз $(\mathbf{r}_1 + \mathbf{r}_2)$. Тогава
  $$\mathcal{L}(\mathbf{r}_1 + \mathbf{r}_2) \stackrel{\text{деф}}{=} \mathcal{L}(\mathbf{r}_1) \cup \mathcal{L}(\mathbf{r}_2).$$
  - $L_1 \cdot L_2$ е регулярен език, който се описва с регулярния израз $(\mathbf{r}_1 \cdot \mathbf{r}_2)$. Тогава Тази операция се нарича конкатенация. Обикновено изпускаме знака $\cdot$
  $$\mathcal{L}(\mathbf{r}_1 \cdot \mathbf{r}_2) \stackrel{\text{деф}}{=} \mathcal{L}(\mathbf{r}_1) \cdot \mathcal{L}(\mathbf{r}_2).$$
  - $L_1^\star$ е регулярен език, който се описва с регулярния израз $\mathbf{r}_1^\star$. Тогава Звезда на Клини
  $$\mathcal{L}(\mathbf{r}_1^\star) \stackrel{\text{деф}}{=} \mathcal{L}(\mathbf{r}_1)^\star.$$

Забележка. Ние знаем, че:

- Всеки регулярен израз представлява крайна дума над крайна азбука. Това означава, че множеството от всички регулярни изрази е изброимо безкрайно. Оттук следва, че всички регулярни езици образуват изброимо безкрайно множество.
- Понеже Σ е крайна азбука, то $\Sigma^\star$ е изброимо безкрайно множество;
- Един език над азбуката Σ представлява елемент на $\mathscr{P}(\Sigma^\star)$. Това означава, че всички езици над азбуката Σ представляват неизброимо безкрайно множество.

От всичко това следва, че има езици, които не са регулярни. По-нататък ще видим примери за такива езици.

В [Sip12, стр. 70] е показан алгоритъм, за който по един автомат може да се получи регулярен израз описващ езика на автомата. Ние няма да разглеждаме този алгоритъм.

Пример 2.2. Нека да построим регулярни изрази за всеки от езиците от *Пример 2.1*.

а) Нека $\mathbf{r} \overset{\text{деф}}{=} (\mathbf{a}+\mathbf{b})^\star\mathbf{bab}(\mathbf{a}+\mathbf{b})^\star$. Тогава

$$\mathcal{L}(\mathbf{r}) = \{\omega \in \{a,b\}^\star \mid \omega \text{ съдържа } bab\}.$$

б) Нека $\mathbf{r} \overset{\text{деф}}{=} \mathbf{b}^\star\mathbf{ab}^\star\mathbf{a}(\mathbf{a}+\mathbf{b})^\star$. Тогава

$$\mathcal{L}(\mathbf{r}) = \{\omega \in \{a,b\}^\star \mid |\omega|_a \geq 2\}.$$

За момента не е ясно как можем да верифицираме дали регулярният израз $\mathbf{r}$ наистина описва посочения език.

в) Нека $\mathbf{r} \overset{\text{деф}}{=} (\mathbf{b}+\mathbf{ab})^\star$. Тогава

$$\mathcal{L}(\mathbf{r}) = \{\omega \in \{a,b\}^\star \mid \text{ всяко } a \text{ в } \omega \text{ се следва от поне едно } b\}.$$

г) Нека $\mathbf{r} \overset{\text{деф}}{=} (\mathbf{b}^\star\mathbf{ab}^\star\mathbf{ab}^\star\mathbf{ab}^\star)^\star$. Тогава

$$\mathcal{L}(\mathbf{r}) = \{\omega \in \{a,b\}^\star \mid |\omega|_a \equiv 0 \bmod 3\}.$$

Задача 2.5. За произволни регулярни изрази $\boldsymbol{r}$ и $\boldsymbol{s}$, ще пишем $\boldsymbol{r} \equiv \boldsymbol{s}$, ако $\mathcal{L}(\mathbf{r}) = \mathcal{L}(\mathbf{s})$. Проверете дали са изпълнени следните равенства:

а) $\boldsymbol{r}+\boldsymbol{s} \equiv \boldsymbol{s}+\boldsymbol{r}$;

б) $(\boldsymbol{\varepsilon}+\boldsymbol{r})^\star \equiv \boldsymbol{r}^\star$;

в) $\boldsymbol{\emptyset}^\star \equiv \boldsymbol{\varepsilon}$;

г) $(\boldsymbol{r}^\star\boldsymbol{s}^\star)^\star \equiv (\boldsymbol{r}+\boldsymbol{s})^\star$;

д) $(\boldsymbol{r}^\star)^\star \equiv \boldsymbol{r}^\star$;

е) $(\boldsymbol{rs}+\boldsymbol{r})^\star\boldsymbol{r} \equiv \boldsymbol{r}(\boldsymbol{sr}+\boldsymbol{r})^\star$;

ж) $\boldsymbol{s}(\boldsymbol{rs}+\boldsymbol{s})^\star\boldsymbol{r} \equiv \boldsymbol{rr}^\star\boldsymbol{s}(\boldsymbol{rr}^\star\boldsymbol{s})^\star$;

з) $(\boldsymbol{r}+\boldsymbol{s})^\star \equiv \boldsymbol{r}^\star+\boldsymbol{s}^\star$;

и) $(\boldsymbol{r}+\boldsymbol{s})^\star\boldsymbol{s} \equiv (\boldsymbol{r}^\star\boldsymbol{s})^\star$;

к) $(\boldsymbol{rs}+\boldsymbol{r})^\star\boldsymbol{rs} \equiv (\boldsymbol{rr}^\star\boldsymbol{s})^\star$;

л) $\boldsymbol{\emptyset}^\star \equiv \boldsymbol{\varepsilon}^\star$.

2.2.1 Всеки автоматен език е регулярен

Нашата цел е да докажем, че автоматните езици съвпадат с регулярните. За целта, ще разгледаме двете посоки на тази задача поотделно. Първо ще докажем, че всеки автоматен език е регулярен, а после ще видим, че всеки регулярен език е автоматен.

Теорема 2.1 (Клини [Kle56]). Всеки автоматен език се описва с регулярен израз.

Доказателство. Нека $L = \mathcal{L}(\mathcal{A})$, за някой детерминиран краен автомат $\mathcal{A}$. Да фиксираме едно изброяване на състоянията $Q = \{q_0, \ldots, q_{n-1}\}$, като $q_{\texttt{start}} = q_0$. Ще означаваме с $L(i,j,k)$ множеството от тези думи, които могат да се разпознаят от автомата по път, който започва от q_i, завършва в q_j, и междинните състояния имат индекси $< k$. Например, за думата $\alpha = a_1 a_2 \cdots a_s$ имаме, че $\alpha \in L(i,j,k)$ точно тогава, когато съществуват състояния $q_{\ell_1}, \ldots, q_{\ell_{s-1}}$, като $\ell_1, \ldots, \ell_{s-1} < k$ и

[PL98, стр. 79]; [HU79, стр. 33]

$$q_i \xrightarrow{a_1} q_{\ell_1} \xrightarrow{a_2} q_{\ell_2} \xrightarrow{a_3} \ldots \xrightarrow{a_{s-1}} q_{\ell_{s-1}} \xrightarrow{a_s} q_j.$$

Тогава, за $n = |Q|$, имаме $L(i,j,n) = \{\alpha \in \Sigma^\star \mid \delta^\star(q_i, \alpha) = q_j\}$. Така получаваме, че

$$\mathcal{L}(\mathcal{A}) = \bigcup\{L(0,j,n) \mid q_j \in F\} = \bigcup_{q_j \in F} L(0,j,n).$$

Ще докажем с *индукция по* k, че $(\forall k \leq n)P(k)$, където

$\mathbf{r}^k_{i,j}$ са регулярни изрази.

$$P(k) \stackrel{\text{деф}}{=} (\forall i < n)(\forall j < n)[L(i,j,k) = \mathcal{L}(\mathbf{r}^k_{i,j})].$$

а) Нека $k = 0$. Ще докажем, че за всеки две състояния $q_i, q_j \in Q$, езикът $L(i,j,0)$ се описва с регулярен израз. Имаме да разгледаме два случая.

- Ако $i = j$, то
$$L(i,j,0) = \{\varepsilon\} \cup \{a \in \Sigma \mid \delta(q_i, a) = q_j\}. \tag{2.4}$$
- Ако $i \neq j$, то
$$L(i,j,0) = \{a \in \Sigma \mid \delta(q_i, a) = q_j\}.$$

И в двата случая, понеже $L(i,j,0)$ е краен език, то е ясно, че той се описва с регулярен израз. Така доказахме $P(0)$.

б) Да предположим, че за всяко $q_i, q_j \in Q$ имаме регулярните изрази $\mathbf{r}^k_{i,j}$, които описват езиците $L(i,j,k)$, т.е. имаме индукционното предположение $P(k)$, за което
$$(\forall i < n)(\forall j < n)[L(i,j,k) = \mathcal{L}(\mathbf{r}^k_{i,j})].$$
Ще докажем $P(k+1)$, т.е. съществуват регулярни изрази $\mathbf{r}^{k+1}_{i,j}$, такива че
$$(\forall i < n)(\forall j < n)[L(i,j,k+1) = \mathcal{L}(\mathbf{r}^{k+1}_{i,j})].$$
За всяка дума $\alpha \in L(i,j,k+1)$ имаме един от следните случаи:

- Състоянието q_k не е измежду вътрешните състояния, които участват в изчислението на автомата $\mathcal{A}$ върху думата α, т.е. имаме следната ситуация:

$$q_i \xrightarrow{a_1} q_{\ell_1} \xrightarrow{a_2} q_{\ell_2} \xrightarrow{a_3} \ldots \xrightarrow{a_{s-1}} q_{\ell_{s-1}} \xrightarrow{a_s} q_j,$$

  където $\ell_1, \ldots, \ell_{s-1} < k$. Това означава, че $\alpha \in L(i, j, k)$.

- Състоянието q_k е измежду вътрешните състояния, които участват в изчислението на автомата $\mathcal{A}$ върху думата α. Ако състоянието q_k се среща да кажем m пъти, то можем да представим думата α като $\alpha = \alpha_1\alpha_2 \cdots \alpha_{m+1}$ и имаме следната ситуация:

$$q_i \to \overset{\alpha_1}{\cdots} \to q_k \to \overset{\alpha_2}{\cdots} \to q_k \to \cdots \to q_k \to \overset{\alpha_{m+1}}{\cdots} \to q_j,$$

  За всяко $\ell = 1, \ldots, m+1$, състоянието q_k не е измежду вътрешните състояния, които участват в изчислението на автомата $\mathcal{A}$ върху α_ℓ, т.е. индексите на всички вътрешни състояния са $< k$. Това означава, че $\alpha_1 \in L(i, k, k)$, $\alpha_\ell \in L(k, k, k)$ за $\ell = 2, \ldots, m$, и $\alpha_{m+1} \in L(k, j, k)$, т.е.

Възможно е $m = 1$. Тогава $L(k, k, k)^{m-1} = \{\varepsilon\}$.

$$\alpha \in L(i, k, k) \cdot L(k, k, k)^{m-1} \cdot L(k, j, k).$$

  Понеже направихме това разсъждение за произволно m, можем да заключим, че ако q_k се среща измежду вътрешните състояния, които участва в изчислението на автомата $\mathcal{A}$ върху думата α, то

$$\alpha \in L(i, k, k) \cdot L(k, k, k)^\star \cdot L(k, j, k).$$

Така доказахме, че

$$L(i, j, k+1) \subseteq L(i, j, k) \cup L(i, k, k) \cdot L(k, k, k)^\star \cdot L(k, j, k).$$

✍ Защо?

Включването в другата посока следва директно от дефиницията на езиците $L(i, j, k)$. От направените по-горе разсъждения следва, че можем да изразим $L(i, j, k+1)$ по следния начин:

$$L(i, j, k+1) = \underbrace{L(i, j, k)}_{\mathcal{L}(\mathbf{r}^k_{i,j})} \cup \underbrace{L(i, k, k)}_{\mathcal{L}(\mathbf{r}^k_{i,k})} \cdot \underbrace{(L(k, k, k))^\star}_{\mathcal{L}(\mathbf{r}^k_{k,k})} \cdot \underbrace{L(k, j, k)}_{\mathcal{L}(\mathbf{r}^k_{k,j})}.$$

Тогава по **И.П.** следва, че $L(i, j, k+1)$ може да се опише с регулярния израз

$$\mathbf{r}^{k+1}_{i,j} = \mathbf{r}^k_{i,j} + \mathbf{r}^k_{i,k} \cdot (\mathbf{r}^k_{k,k})^\star \cdot \mathbf{r}^k_{k,j}. \tag{2.5}$$

Така доказахме $P(k+1)$.

Заключаваме, че за всеки два индекса $i, j < |Q|$ и индекс $k \leq |Q|$, езикът $L(i, j, k)$ може да се опише с регулярен израз $\mathbf{r}^k_{i,j}$. Тогава ако $F = \{q_{i_1}, \ldots, q_{i_m}\}$, то $\mathcal{L}(\mathcal{A})$ се описва с регулярния израз

$$\mathbf{r}^n_{0,i_1} + \mathbf{r}^n_{0,i_2} + \cdots + \mathbf{r}^n_{0,i_m}.$$

□

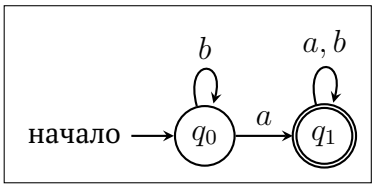
Твърдение 2.5. Съществува алгоритъм, за който при вход краен детерминиран автомат $\mathcal{A}$, извежда като изход регулярен израз $\mathbf{r}$, такъв че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbf{r})$.

Естествено, можем да формулираме това твърдение и за краен недетерминиран автомат.

Доказателството на това твърдение използва идеята от доказателството на теоремата на Клини но излиза извън рамките на този курс. За подробно изложение на този въпрос, вижте [Sip12, стр. 69]. Възможно е по-късно да използваме този резултат наготово. Нека поне да разгледаме примери, чиято цел е да ни убеди, че наистина може да се извлече алгоритъм от доказателството на теоремата на Клини.

Примерни задачи

Задача 2.6. Приложете конструкцията от теоремата на Клини за да получите регулярен израз, който описва езика на автомата $\mathcal{A}$ изобразен на Фигура 2.7.



Фигура 2.7: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}(\mathbf{b}^\star\mathbf{a}(\mathbf{a}+\mathbf{b})^\star)$

Решение. Лесно се съобразява, че езикът на автомата от *Фигура 2.7* се описва с регулярния израз $\mathbf{b}^\star\mathbf{a}(\mathbf{a}+\mathbf{b})^\star$. Следвайки означенията и конструкцията от доказателството на теоремата на Клини, езикът на този автомат се описва с регулярния израз $\mathbf{r}^2_{0,1}$, защото началното състояние е q_0, финалното е q_1 и броят на състоянията в автомата е 2. Подробните сметки са следните:

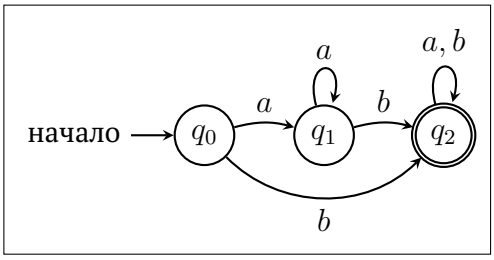
$$
\begin{aligned}
r^2_{0,1} &\equiv \underbrace{r^1_{0,1}}_{b^\star a} + \underbrace{r^1_{0,1}}_{b^\star a} \cdot \underbrace{(r^1_{1,1})^\star}_{(\varepsilon+\mathbf{a}+\mathbf{b})^\star} \cdot \underbrace{\mathbf{r}^1_{1,1}}_{\varepsilon+\mathbf{a}+\mathbf{b}} && \text{// според (2.5)} \\
&\equiv \mathbf{b}^\star\mathbf{a} + \mathbf{b}^\star\mathbf{a}(\varepsilon+\mathbf{a}+\mathbf{b})^\star(\varepsilon+\mathbf{a}+\mathbf{b}) && \text{// просто заместваме} \\
&\equiv \mathbf{b}^\star\mathbf{a} + \mathbf{b}^\star\mathbf{a}(\varepsilon+\mathbf{a}+\mathbf{b})^+ && \text{// } \mathbf{r}^+ \stackrel{\text{деф}}{=} \mathbf{r}^\star\mathbf{r} \\
&\equiv \mathbf{b}^\star\mathbf{a} + \mathbf{b}^\star\mathbf{a}(\mathbf{a}+\mathbf{b})^\star && \text{// } \mathbf{r}^\star \equiv (\varepsilon+\mathbf{r})^+ \\
&\equiv \mathbf{b}^\star\mathbf{a}(\varepsilon + (\mathbf{a}+\mathbf{b})^\star) && \text{// } \mathbf{r}+\mathbf{rq} \equiv \mathbf{r}(\varepsilon+\mathbf{q}) \\
&\equiv \mathbf{b}^\star\mathbf{a}(\mathbf{a}+\mathbf{b})^\star. && \text{// } \mathbf{r}^\star \equiv \varepsilon + \mathbf{r}^\star
\end{aligned}
$$

Тук използвахме, че:

$$
\begin{aligned}
\mathbf{r}^1_{\mathbf{0,1}} &\equiv \underbrace{\mathbf{r}^0_{0,1}}_{\mathbf{a}} + \underbrace{\mathbf{r}^0_{0,0}}_{\varepsilon+b} \cdot \underbrace{(\mathbf{r}^0_{\mathbf{0,0}})^\star}_{(\varepsilon+b)^\star} \cdot \underbrace{\mathbf{r}^0_{\mathbf{0,1}}}_{\mathbf{b}} && \text{// според (2.5)} \\
&\equiv a + (\varepsilon+b)(\varepsilon+b)^\star a && \text{// просто заместваме} \\
&\equiv \mathbf{a} + \mathbf{b}^\star\mathbf{a} && \text{// } \mathbf{r}^\star \equiv \varepsilon + \mathbf{r}^\star \\
&\equiv \mathbf{b}^\star\mathbf{a} \\
\mathbf{r}^b_{b,b} &\equiv \underbrace{\mathbf{r}^a_{b,b}}_{\varepsilon+a+b} + \underbrace{\mathbf{r}^a_{b,a}}_{\emptyset} \cdot \underbrace{(\mathbf{r}^a_{a,a})^\star}_{\varepsilon+b} \cdot \underbrace{\mathbf{r}^a_{a,b}}_{a} && \text{// отново според (2.5)} \\
&\equiv \varepsilon + a + b + \emptyset(\varepsilon+b)^\star a && \text{// просто заместваме} \\
&\equiv \varepsilon + a + b. && \text{// защото } \emptyset \cdot r \equiv \emptyset
\end{aligned}
$$

□

Задача 2.7. Приложете конструкцията от теоремата на Клини за да получите регулярен израз, който описва езика на автомата $\mathcal{A}$ изобразен на Фигура 2.8.



Фигура 2.8: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}(\mathbf{a^\star b(a+b)^\star})$.

Решение. От теоремата на Клини знаем, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbf{r}^3_{0,2})$, където:

$$
\begin{aligned}
\mathbf{r}^3_{0,2} &\equiv \underbrace{\mathbf{r}^2_{0,2}}_{\boldsymbol{a^\star b}} + \underbrace{\mathbf{r}^2_{0,2}}_{\boldsymbol{a^\star b}} \cdot \underbrace{(\mathbf{r}^2_{2,2})^\star}_{\boldsymbol{(\varepsilon+a+b)^\star}} \cdot \underbrace{\mathbf{r}^2_{2,2}}_{\boldsymbol{\varepsilon+a+b}} && \text{// според (2.5)} \\
&\equiv \boldsymbol{a^\star b + a^\star b \cdot (a+b)^\star \cdot (\varepsilon + a + b)} && \text{// просто заместваме} \\
&\equiv \boldsymbol{a^\star b + a^\star b \cdot (a+b)^\star} && \text{// } \boldsymbol{r^\star \equiv r^\star \cdot (\varepsilon + r)} \\
&\equiv \boldsymbol{a^\star b(\varepsilon + (a+b)^\star)} && \text{// } \boldsymbol{r_1 + r_1 \cdot r_2 \equiv r_1 \cdot (\varepsilon + r_2)} \\
&\equiv \boldsymbol{a^\star b(a+b)^\star}. && \text{// } \boldsymbol{r^\star \equiv \varepsilon + r^\star}
\end{aligned}
$$

Тук използвахме, че:

$$
\begin{aligned}
\mathbf{r}^2_{0,2} &\equiv \underbrace{\mathbf{r}^1_{0,2}}_{\boldsymbol{b}} + \underbrace{\mathbf{r}^1_{0,1}}_{\boldsymbol{a}} \cdot \underbrace{(\mathbf{r}^1_{1,1})^\star}_{\boldsymbol{a}} \cdot \underbrace{\mathbf{r}^1_{1,2}}_{\boldsymbol{b}} && \text{// според (2.5)} \\
&\equiv \boldsymbol{b + a \cdot a^\star \cdot b} && \text{// просто заместваме} \\
&\equiv \boldsymbol{(\varepsilon + a^+) \cdot b} && \text{// } \mathbf{r}^+ \stackrel{\text{деф}}{=} \mathbf{r} \cdot \mathbf{r}^\star \\
&\equiv \boldsymbol{a^\star b}. && \text{// } \mathbf{r}^\star \equiv \varepsilon + \boldsymbol{r^+}.
\end{aligned}
$$

Заключаваме, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbf{a^\star b(a+b)^\star})$. □

2.3 Недетерминирани крайни автомати

Определение 2.2. Недетерминиран краен автомат представлява петорка

$$\mathcal{N} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle,$$

Въведени от Рабин и Скот [RS59]. За по-голяма яснота, често ще означаваме с $\mathcal{N}$ недетерминирани автомати, като ще запазим $\mathcal{A}$ за детерминираните. Мотивация - [Koz97, стр. 25].

- Q е крайно множество от състояния;
- Σ е крайна азбука;
- $\Delta : Q \times \Sigma \to \mathscr{P}(Q)$ е функцията на преходите. Да обърнем внимание, че е възможно за някоя двойка (q, a) да няма нито един преход в автомата. Това е възможно, когато $\Delta(q, a) = \emptyset$;

Да напомним, че $\mathscr{P}(Q) \stackrel{\text{деф}}{=} \{R \mid R \subseteq Q\}$, $|\mathscr{P}(Q)| = 2^{|Q|}$

- $Q_{\texttt{start}} \subseteq Q$ е множество от начални състояния;
- $F \subseteq Q$ е множеството от финални състояния.

Удобно е да разширим функцията на преходите $\Delta : Q \times \Sigma \to \mathscr{P}(Q)$ до функцията $\Delta^\star : \mathscr{P}(Q) \times \Sigma^\star \to \mathscr{P}(Q)$, която дефинираме за произволно $R \subseteq Q$ и $\alpha \in \Sigma^\star$ по следния начин:

В [PL98] Δ е релация и се позволяват ε-преходи. В [Sip12] пък е функция, но пак се позволяват ε-преходи. В [HU79] е функция без ε-преходи. Навсякъде има само едно начално.

- Ако $\alpha = \varepsilon$, то $\Delta^\star(R, \varepsilon) \stackrel{\text{деф}}{=} R$;
- Ако $\alpha = \beta a$, то $\Delta^\star(R, \beta a) \stackrel{\text{деф}}{=} \bigcup\{\Delta(p, a) \mid p \in \Delta^\star(R, \beta)\}$.

Да напомним, че $\bigcup\{\{0,1\},\{1,2,3\}\} = \{0,1\} \cup \{1,2,3\}$.

$$\mathcal{L}(\mathcal{N}) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \Delta^\star(Q_{\texttt{start}}, \omega) \cap F \neq \emptyset\}.$$

Твърдение 2.6. За всеки две думи $\alpha, \beta \in \Sigma^\star$ и всяко $R \subseteq Q$,

Сравенете с *Твърдение 2.1*.

$$\Delta^\star(R, \alpha\beta) = \Delta^\star(\Delta^\star(R, \alpha), \beta).$$

Доказателство. Индукция по дължината на β.

- Нека $|\beta| = 0$, т.е. $\beta = \varepsilon$. Тогава:

$$\begin{aligned} \Delta^\star(R, \alpha\varepsilon) &= \Delta^\star(R, \alpha) && \text{// } \alpha\varepsilon = \alpha \\ &= \Delta^\star(\Delta^\star(R, \alpha), \varepsilon). && \text{// деф. на } \Delta^\star \end{aligned}$$

- Да приемем, че твърдението е вярно за думи β с дължина n.

- Нека $|\beta| = n+1$, т.е. $\beta = \gamma b$, където $|\gamma| = n$.

$$\begin{aligned}
\Delta^\star(R, \alpha\gamma b) &= \bigcup\{\Delta(p,b) \mid p \in \Delta^\star(R, \alpha\gamma)\} && \text{// от деф. на } \Delta^\star \\
&= \bigcup\{\Delta(p,b) \mid p \in \Delta^\star(\underbrace{\Delta^\star(R,\alpha)}_{U}, \gamma))\} && \text{// от И.П. за } \gamma \\
&= \bigcup\{\Delta(p,b) \mid p \in \Delta^\star(U, \gamma)\} && \text{// нека } U \stackrel{\text{деф}}{=} \Delta^\star(R,\alpha) \\
&= \Delta^\star(U, \gamma b) && \text{// от деф. на } \Delta^\star \\
&= \Delta^\star(\Delta^\star(R,\alpha), \gamma b) && \text{// } U = \Delta^\star(R,\alpha)
\end{aligned}$$

□

И тук е удобно да въведем бинарната релация $\vdash_{\mathcal{N}}$ над $Q \times \Sigma^\star$, която ще ни казва как текущото състояние (конфигурацията) на автомата $\mathcal{N}$ се променя след изпълнение на една стъпка:

$$\frac{p \in \Delta(q,a)}{(q, a\beta) \vdash_{\mathcal{N}} (p, \beta)}$$

Фигура 2.9: Едностъпков преход в недетерминиран краен автомат $\mathcal{N}$

Рефл. и транз. затваряне на една релация е разгледано в Глава 1. Тук $\vdash^\star_{\mathcal{N}}$ е рефлексивното и транзитивно затваряне на релацията $\vdash_{\mathcal{N}}$.

Ще дефинираме релацията $\vdash^\ell_{\mathcal{N}}$, която определя работата на автомата $\mathcal{N}$ за ℓ на брой стъпки.

$$\frac{}{\kappa \vdash^0_{\mathcal{N}} \kappa} \qquad \frac{\kappa \vdash_{\mathcal{N}} \kappa'' \qquad \kappa'' \vdash^\ell_{\mathcal{N}} \kappa'}{\kappa \vdash^{\ell+1}_{\mathcal{N}} \kappa'}$$

Сега можем да дефинираме $\vdash^\star_{\mathcal{N}}$ като:

$$(q,\alpha) \vdash^\star_{\mathcal{N}} (p,\beta) \stackrel{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[(q,\alpha) \vdash^\ell_{\mathcal{N}} (p,\beta)].$$

Друг начин да дефинираме релацията $\vdash^\star_{\mathcal{N}}$ е следния: $(q,\alpha\beta) \vdash^\star_{\mathcal{N}} (p,\beta)$ точно тогава, когато $p \in \Delta^\star(\{q\}, \alpha)$.

Получаваме, че

$$\mathcal{L}(\mathcal{N}) = \{\alpha \in \Sigma^\star \mid (\exists q \in Q_{\texttt{start}})(\exists f \in F)[(q,\alpha) \vdash^\star_{\mathcal{N}} (f,\varepsilon)]\}.$$

Теорема 2.2 (Рабин-Скот [RS59]). За всеки недетерминиран краен автомат $\mathcal{N}$ съществува еквивалентен на него детерминиран краен автомат $\mathcal{D}$, т.е.

$$\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{D}).$$

Упътване. Нека $\mathcal{N} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle$. Ще построим детерминиран автомат

$$\mathcal{D} = (Q', \Sigma, \delta, q_{\texttt{start}}, F'),$$

за който $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{D})$. Конструкцията е следната:

- $Q' = \{\ulcorner R \urcorner \mid R \subseteq Q\}$;
- За произволна буква $a \in \Sigma$ и произволно $R \subseteq Q$,

$$\delta(\ulcorner R \urcorner, a) \stackrel{\text{деф}}{=} \ulcorner \Delta^\star(R, a) \urcorner.$$

- $q_{\texttt{start}} = \ulcorner Q_{\texttt{start}} \urcorner$;
- $F' \stackrel{\text{деф}}{=} \{\ulcorner R \urcorner \in Q' \mid R \cap F \neq \emptyset\}$.

Да отбележим, че детерминираният автомат $\mathcal{D}$ има не повече от $2^{|Q|}$ на брой състояния. Реално на нас ни трябват само тези множества R, за които съществува дума α и $\Delta^\star(Q_{\texttt{start}}, \alpha) = R$.

Ще докажем, че за произволна дума α и произволно множество $R \subseteq Q$ е изпълнено, че:

$$\ulcorner \Delta^\star(R, \alpha) \urcorner = \delta^\star(\ulcorner R \urcorner, \alpha). \tag{2.6}$$

Тук използваме $\ulcorner R \urcorner$ за да покажем в явен вид, че в новия автомат кодираме множества от състояния на стария.

Това ще направим с индукция по дължината на думата α.

- Ако $|\alpha| = 0$, т.е. $\alpha = \varepsilon$, то е ясно от дефиницията на $\Delta^\star$ и $\delta^\star$, т.е. за всяко $R \subseteq Q$ е изпълнено, че:
$$\ulcorner \Delta^\star(R, \varepsilon) \urcorner = \ulcorner R \urcorner = \delta^\star(\ulcorner R \urcorner, \varepsilon).$$
- Да приемем, че (2.6) е изпълнено за думи α с дължина n, т.е.
$$(\forall \alpha \in \Sigma^n)(\forall R \subseteq Q)[\ \ulcorner \Delta^\star(R, \alpha) \urcorner = \delta^\star(\ulcorner R \urcorner, \alpha)\ ].$$
- Нека сега α има дължина $n+1$, т.е. $\alpha = \beta a$, където $|\beta| = n$ и $a \in \Sigma$.

$$\begin{aligned}
\delta^\star(\ulcorner R \urcorner, \beta a) &= \delta(\delta^\star(\ulcorner R \urcorner, \beta a)) && \text{// деф. на } \delta^\star \\
&= \delta(\ulcorner \Delta^\star(R, \beta) \urcorner, a) && \text{// от И.П. за } \beta \\
&= \ulcorner \Delta^\star(\Delta^\star(R, \beta), a) \urcorner && \text{// от деф. на } \delta \\
&= \ulcorner \Delta^\star(R, \beta a) \urcorner. && \text{// от } \textit{Твърдение 2.6}
\end{aligned}$$

Така доказахме, че

$$(\forall \alpha \in \Sigma^{n+1})(\forall R \subseteq Q)[\ \ulcorner \Delta^\star(R, \alpha) \urcorner = \delta^\star(\ulcorner R \urcorner, \alpha)\ ].$$

Това означава, че според принципа на математическата индукция имаме Свойство (2.6).

Сега вече е лесно да съобразим, че

$$\begin{aligned}
\omega \in \mathcal{L}(\mathcal{D}) &\Leftrightarrow \delta^\star(q_{\texttt{start}}, \omega) \in F' && \text{// деф. на } \mathcal{L}(\mathcal{D}) \\
&\Leftrightarrow \Delta^\star(Q_{\texttt{start}}, \omega) \cap F \neq \emptyset && \text{// от (2.6)} \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{N}) && \text{// деф. на } \mathcal{L}(\mathcal{N}).
\end{aligned}$$

□

Хубаво е да има един пример за детерминизация.

По-късно ще видим, че можем да дадем и друго доказателство на това твърдение, като направим индукция по построението на регулярните езици.

Задача 2.8. Докажете, че автоматните езици са затворени относно операцията `rev`. С други думи, докажете, че ако L е автоматен език, то $L^{\texttt{rev}} \stackrel{\text{деф}}{=} \{\omega^{\texttt{rev}} \mid \omega \in L\}$ също е автоматен.

Упътване. Идеята е съвсем проста - просто обръщаме стрелките и правим финалните състояния да са начални, а началното става финално. Нека $L = \mathcal{L}(\mathcal{A})$. Ще построим НКА $\mathcal{N}$, за който $\mathcal{L}(\mathcal{N}) = L^{\texttt{rev}}$.

- $Q^{\mathcal{N}} = Q^{\mathcal{A}}$;
- $Q_{\texttt{start}} = F^{\mathcal{A}}$;
- $\Delta_{\mathcal{N}}(q, a) = \{p \in Q^{\mathcal{A}} \mid \delta_{\mathcal{A}}(p, a) = q\}$;
- $F^{\mathcal{N}} = \{q_{\texttt{start}}\}$.

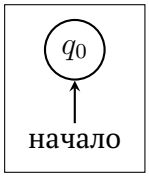
Достатъчно е да се докаже, че $\Delta^{\star}_{\mathcal{N}}(Q_{\texttt{start}}, \alpha) = \{q \in Q^{\mathcal{A}} \mid \delta^{\star}_{\mathcal{A}}(q, \alpha^{\texttt{rev}}) \in F^{\mathcal{A}}\}$. □

Лема 2.1. Езикът L е автоматен, където:

- $L = \emptyset$,
- $L = \{\varepsilon\}$, или
- $L = \{a\}$, за произволна буква $a \in \Sigma$.

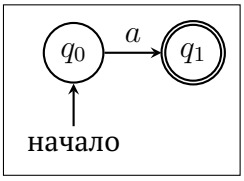
Упътване. Достатъчно е да покажем, че съществуват недетерминирани крайни автомати $\mathcal{N}$.

Според нашата дефиниция, тук описваме недетерминирани автомати, защото за да бъдат детерминирани, трябва функцията на преходите да бъде тотална.



(а) $L(\mathcal{N}) = \emptyset$

(б) $\mathcal{L}(\mathcal{N}) = \{\varepsilon\}$



(в) $\mathcal{L}(\mathcal{N}) = \{a\}$

□

2.3.1 Затвореност относно регулярните операции

Лема 2.2. Класът на автоматните езици е затворен относно операцията *конкатенация*. Това означава, че ако L_1 и L_2 са два произволни автоматни езика, то $L_1 \cdot L_2$ също е автоматен език.

Тук отново приемаме, че $Q_1 \cap Q_2 = \emptyset$.

Доказателство. Нека за по-просто да вземем два детерминирани крайни автомата:

- $\mathcal{A}_1 = \langle \Sigma, Q_1, \delta_1, q'_{\texttt{start}}, F_1 \rangle$, където $\mathcal{L}(\mathcal{A}_1) = L_1$;

- $\mathcal{A}_2 = \langle \Sigma, Q_2, \delta_2, q''_{\texttt{start}}, F_2 \rangle$, където $\mathcal{L}(\mathcal{A}_2) = L_2$.

Ще дефинираме автомата $\mathcal{N} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle$ по такъв начин, че

$$\mathcal{L}(\mathcal{N}) = L_1 \cdot L_2 = \mathcal{L}(\mathcal{A}_1) \cdot \mathcal{L}(\mathcal{A}_2).$$

- $Q \stackrel{\text{деф}}{=} Q_1 \cup Q_2$;
- $Q_{\texttt{start}} \stackrel{\text{деф}}{=} \{q'_{\texttt{start}}\}$;
- $F \stackrel{\text{деф}}{=} \begin{cases} F_1 \cup F_2, & \text{ако } q''_{\texttt{start}} \in F_2 \\ F_2, & \text{иначе.} \end{cases}$
- $\Delta(q,a) \stackrel{\text{деф}}{=} \begin{cases} \{\delta_1(q,a)\}, & \text{ако } q \in Q_1 \setminus F_1 \ \& \ a \in \Sigma \\ \{\delta_1(q,a), \delta_2(q''_{\texttt{start}}, a)\}, & \text{ако } q \in F_1 \ \& \ a \in \Sigma \\ \{\delta_2(q,a)\}, & \text{ако } q \in Q_2 \ \& \ a \in \Sigma. \end{cases}$

Първо ще докажем, че

$$\mathcal{L}(\mathcal{A}_1) \cdot \mathcal{L}(\mathcal{A}_2) \subseteq \mathcal{L}(\mathcal{N}).$$

За целта, нека разгледаме думата $\alpha \in \mathcal{L}(\mathcal{A}_1)$ и $\beta \in \mathcal{L}(\mathcal{A}_2)$. Това означава, че имаме следните изчисления:

$$(q'_{\texttt{start}}, \alpha) \vdash^\star_{\mathcal{A}_1} (q_1, \varepsilon), \text{ за някое } q_1 \in F_1$$
$$(q''_{\texttt{start}}, \beta) \vdash^\star_{\mathcal{A}_2} (q_2, \varepsilon), \text{ за някое } q_2 \in F_2.$$

Според дефиницията на недетерминирания краен автомат $\mathcal{N}$ е ясно, че:

$$(q'_{\texttt{start}}, \alpha) \vdash^\star_{\mathcal{N}} (q_1, \varepsilon), \text{ за някое } q_1 \in F_1.$$

Ако $\beta = \varepsilon$, то това означава, че $q''_{\texttt{start}} \in F_2$ и следователно $F_1 \subseteq F$. Тогава получаваме, че $\alpha \cdot \beta = \alpha \in \mathcal{L}(\mathcal{N})$, защото

$$(q'_{\texttt{start}}, \alpha) \vdash^\star_{\mathcal{N}} (q_1, \varepsilon), \text{ за някое } q_1 \in F_1 \subseteq F.$$

Ако $\beta = b\gamma$, за някоя дума $\gamma \in \Sigma^\star$, то тогава можем да разбием изчислението на β в $\mathcal{A}_2$ по следния начин:

$$(q''_{\texttt{start}}, b\gamma) \vdash_{\mathcal{A}_2} (q, \gamma) \vdash^\star_{\mathcal{A}_2} (q_2, \varepsilon), \text{ за някое } q_2 \in F_2,$$

където $q = \delta_2(q''_{\texttt{start}}, b)$.

Според дефиницията на недетерминирния краен автомат $\mathcal{N}$ е ясно, че:

$$(q, \gamma) \vdash^\star_{\mathcal{N}} (q_2, \varepsilon), \text{ за някое } q_2 \in F_2.$$

Освен това, имаме, че $q \in \Delta(q_1, b)$, защото $q_1 \in F_1$. Това означава, че

$$(q_1, b\gamma) \vdash_{\mathcal{N}} (q, \gamma).$$

Съединявайки последните две изчисления, получаваме, че:

$$(q_1, \beta) \vdash_{\mathcal{N}}^{\star} (q_2, \varepsilon), \text{ за някое } q_2 \in F_2.$$

Сега съединяваме и изчислението за α и получаваме, че:

$$(q'_{\texttt{start}}, \alpha\beta) \vdash_{\mathcal{N}}^{\star} (q_1, \beta) \vdash_{\mathcal{N}}^{\star} (q_2, \varepsilon), \text{ за някое } q_2 \in F_2.$$

Оттук заключаваме, че във всички случаи за β, $\alpha \cdot \beta \in \mathcal{L}(\mathcal{N})$.

Сега ще докажем, че

$$\mathcal{L}(\mathcal{N}) \subseteq \mathcal{L}(\mathcal{A}_1) \cdot \mathcal{L}(\mathcal{A}_2).$$

За целта, нека разгледаме думата $\omega \in \mathcal{L}(\mathcal{N})$, където $\omega = a_0 a_1 \cdots a_{n-1}$. Да разгледаме една редица от състояния $(q_i)_{i=0}^{n}$, която описва приемащо изчисление на $\mathcal{N}$ върху ω. Това означава, че:

Възможно е да има и други редици от състояния $(p_i)_{i=0}^{n}$, които да описват приемащи изчисления на $\mathcal{N}$ върху ω.

- $q_0 = q_{\texttt{start}}$;
- $q_{i+1} \in \Delta(q_i, \omega\,[i]\,)$ за $i < n$;
- $q_n \in F$.

Ако $q_n \in F_1$, то според конструкцията на $\mathcal{N}$, $\varepsilon \in \mathcal{L}(\mathcal{A}_2)$ и всяко състояние от $(q_i)_{i=0}^{n}$ принадлежи на Q_1 и оттам $\omega \in \mathcal{L}(\mathcal{A}_1)$. Интересният случай е когато $q_n \in F_2$. Според конструкцията на $\mathcal{N}$, не можем да преминем от състояние от Q_2 в състояние от Q_1. Това означава, че можем да разбием редицата от състояния $(q_i)_{i=0}^{n}$ на две непразни подредици:

- $(q_i)_{i=0}^{\ell}$ - тези които са от Q_1,
- $(q_i)_{i=\ell+1}^{n}$ - тези, които са от Q_2.

Нека $\alpha = \omega\,[:\ell\,]$ и $\beta = \omega\,[\ell\,:]$. Ясно е, че:

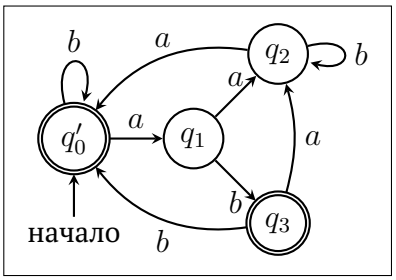
$$(q_0, \omega) \vdash_{\mathcal{N}}^{\star} (q_\ell, \omega\,[\ell\,:]\,) \vdash_{\mathcal{N}} (q_{\ell+1}, \omega\,[\ell+1\,:]\,) \vdash_{\mathcal{N}}^{\star} (q_n, \varepsilon).$$

От конструкцията на $\mathcal{N}$ следва, че редицата от състояния $(q_i)_{i=0}^{\ell}$ описва приемащо изчисление на $\mathcal{A}_1$ върху α. Също така от конструкцията следва, че щом $q_{\ell+1} \in \Delta(q_\ell, a_\ell)$, то $q_\ell \in F_1$ и $\delta_2(q''_{\texttt{start}}, a_\ell) = q_{\ell+1}$. Заключаваме, че:

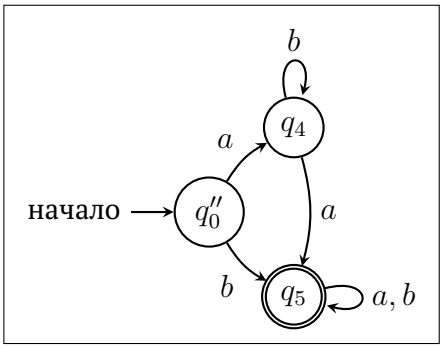
Това е единственият начин да направим преход от състояние на Q_1 към състояние на Q_2.

- $(q_0, \alpha) \vdash_{\mathcal{A}_1}^{\star} (q_\ell, \varepsilon)$. Понеже $q_0 = q'_{\texttt{start}}$ и $q_\ell \in F_1$, то $\alpha \in \mathcal{L}(\mathcal{A}_1)$.
- $(q''_{\texttt{start}}, \beta) \vdash_{\mathcal{A}_2}^{\star} (q_n, \varepsilon)$. Понеже $q_n \in F_2$, то $\beta \in \mathcal{L}(\mathcal{A}_2)$.

□

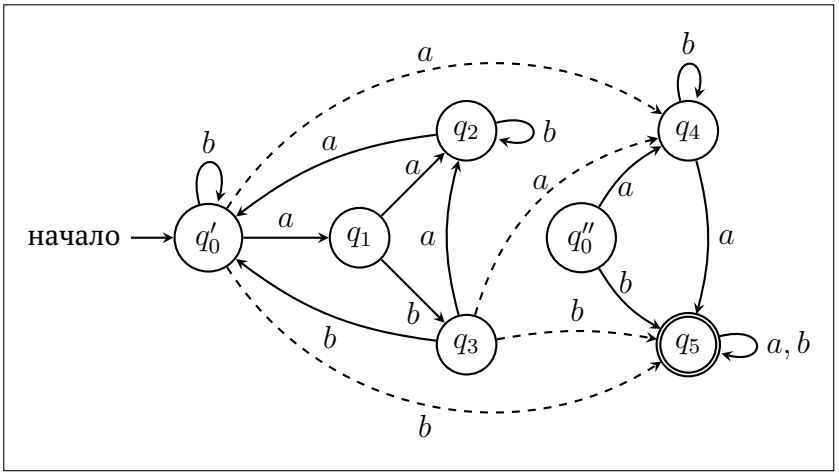


(а) автомат $\mathcal{A}_1$



(б) автомат $\mathcal{A}_2$

Пример 2.3. За да построим автомат, който разпознава конкатенацията на $\mathcal{L}(\mathcal{A}_1)$ и $\mathcal{L}(\mathcal{A}_2)$, трябва да свържем финалните състояния на $\mathcal{A}_1$ с изходящите от s_2 състояния на $\mathcal{A}_2$.



Фигура 2.13: $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A}_1) \cdot \mathcal{L}(\mathcal{A}_2)$

Обърнете внимание, че $\mathcal{A}_1$ и $\mathcal{A}_2$ са детерминирани автомати, но $\mathcal{N}$ е недетерминиран. Също така, в този пример се оказва, че вече q_0'' е недостижимо състояние, но в общия случай не можем да го премахнем, защото може да има преходи влизащи в q_0''.

Лема 2.3. Класът от автоматните езици е затворен относно операцията *обединение*.

Упътване. Нека са дадени детерминистичните автомати:

- $\mathcal{A}_1 = \langle \Sigma, Q_1, \delta_1, q'_{\texttt{start}}, F_1 \rangle$, като $L(\mathcal{A}_1) = L_1$;

Да напомним, че вече знаем, че автоматните езици са затворени относно операцията обединие. Това видяхме в *Твърдение 2.4*. Сега ще дадем втора конструкция.

- $\mathcal{A}_2 = \langle \Sigma, Q_2, \delta_2, q''_{\texttt{start}}, F_2 \rangle$, като $L(\mathcal{A}_2) = L_2$.

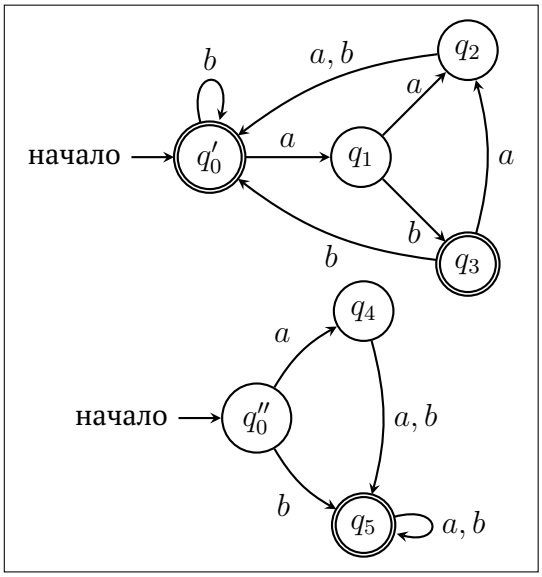
Ще дефинираме автомата $\mathcal{N} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle$, така че

$$L(\mathcal{N}) = L(\mathcal{A}_1) \cup L(\mathcal{A}_2).$$

- $Q_{\texttt{start}} = \{q'_{\texttt{start}}, q''_{\texttt{start}}\}$;
- $Q \stackrel{\text{деф}}{=} Q_1 \cup Q_2$;
- $F \stackrel{\text{деф}}{=} F_1 \cup F_2$;
- $\Delta(q,a) \stackrel{\text{деф}}{=} \begin{cases} \{\delta_1(q,a)\}, & \text{ако } q \in Q_1 \text{ и } a \in \Sigma \\ \{\delta_2(q,a)\}, & \text{ако } q \in Q_2 \text{ и } a \in \Sigma. \end{cases}$

□

Пример 2.4. За да построим автомат, който разпознава обединението на $\mathcal{L}(\mathcal{A}_1)$ и $\mathcal{L}(\mathcal{A}_2)$, трябва да добавим ново начално състояние, което да свържем с наследниците на началните състояния на $\mathcal{A}_1$ и $\mathcal{A}_2$.



Фигура 2.14: $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

Обърнете внимание, че $\mathcal{A}_1$ и $\mathcal{A}_2$ са детерминирани автомати, но $\mathcal{N}$ е недетерминиран. Освен това, новото състояние q_0 трябва да бъде маркирано като финално, защото q'_0 е финално.

Лема 2.4. Класът от автоматните езици е затворен относно операцията *звезда на Клини*, т.е. за всеки автоматен език L, езикът $L^\star$ също е автоматен.

Доказателство. Да разгледаме детерминирния краен автомат

$$\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle.$$

Първо ще построим недетерминиран краен автомат

$$\mathcal{N} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle,$$

такъв че

$$\mathcal{L}(\mathcal{N}) = (\mathcal{L}(\mathcal{A}))^{+}.$$

После ще построим недетерминиран краен автомат $\mathcal{N}'$, за който

$$\mathcal{L}(\mathcal{N}') = \mathcal{L}(\mathcal{N}) \cup \{\varepsilon\} = (\mathcal{L}(\mathcal{A}))^{\star}.$$

Дефинираме функцията на преходите Δ на $\mathcal{N}$ като за $q \in Q$ и $a \in \Sigma$ определяме функцията на преходите Δ по следния начин:

$$\Delta(q,a) \stackrel{\text{деф}}{=} \begin{cases} \{\delta(q,a)\}, & \text{ако } q \notin F \\ \{\delta(q,a), \delta(q_{\texttt{start}}, a)\}, & \text{ако } q \in F. \end{cases}$$

Нека $\alpha = a_0 a_1 \cdots a_{n-1} \in \mathcal{L}(\mathcal{N})$. Това означава, че $(q_{\texttt{start}}, \alpha) \vdash^{\star}_{\mathcal{N}} (f, \varepsilon)$ за някое $f \in F$. Нека редицата от състояния $(q_i)_{i=0}^{n}$ описва едно приемащо изчисление на $\mathcal{N}$ върху α, т.е.

- $q_0 = q_{\texttt{start}}$;
- $q_{i+1} \in \Delta(q_i, a_i)$;
- $q_n \in F$.

Да разгледаме максималната подпоследователност от състояния $(q_{i_j})_{j=0}^{\ell+1}$ на $(q_i)_{i=0}^{n}$ съставена от тези състояния, за които

Тук е малко по-сложно, защото правим разбиване на изчислението не на база всички финални състояния, а на тези финални състояния, от които изчислението продължава в насленик на $q_{\texttt{start}}$, защото това са местата, където можем да разцепим думата на части.

- $q_{i_0} = q_{\texttt{start}}$;
- $q_{i_j} \in F \;\&\; \delta(q_{\texttt{start}}, a_{i_j}) = q_{i_j+1}$, за $j = 1, \ldots, \ell$;
- $q_{i_{\ell+1}} = q_n$.

Да разбием думата α като $\alpha = \alpha_0 \alpha_1 \cdots \alpha_l$, където:

$$\begin{aligned} \alpha_0 &\stackrel{\text{деф}}{=} a_{i_0} \alpha'_0 \\ \alpha_1 &\stackrel{\text{деф}}{=} a_{i_1} \alpha'_1 \\ &\cdots \\ \alpha_\ell &\stackrel{\text{деф}}{=} a_{i_\ell} \alpha'_\ell. \end{aligned}$$

Сега можем да разбием изчислението на $\mathcal{N}$ върху α по следния начин:

$$\begin{aligned}(q_{i_0}, \alpha_0\alpha_1 \cdots \alpha_\ell) &\vdash^\star_{\mathcal{N}} (q_{i_1}, \alpha_1 \cdots \alpha_\ell) && \text{// } q_{i_0} = q_{\texttt{start}} \\ &\vdash^\star_{\mathcal{N}} (q_{i_2}, \alpha_2 \cdots \alpha_\ell) \\ &\cdots \\ &\vdash^\star_{\mathcal{N}} (q_{i_\ell}, \alpha_\ell) \\ &\vdash^\star_{\mathcal{N}} (q_{i_{\ell+1}}, \varepsilon). && \text{// } q_{i_{\ell+1}} = q_n \in F\end{aligned}$$

Да разгледаме само първата част на изчислението:

$$\underbrace{(q_{i_0}, \alpha_0) \vdash^\star_{\mathcal{N}} (q_{i_1}, \varepsilon)}_{\text{само преходи от } \mathcal{A}}.$$

Понеже $q_{i_0} = q_{\texttt{start}}$ и $q_{i_1} \in F$, то е ясно, че $\alpha_0 \in \mathcal{L}(\mathcal{A})$.

За $j = 0, \ldots, \ell$, изчислението

$$(q_{i_j}, \alpha_j) \vdash^\star_{\mathcal{N}} (q_{i_{j+1}}, \varepsilon)$$

може по-подробно да се запише и така:

$$(q_{i_j}, a_{i_j}\alpha'_j) \vdash_{\mathcal{N}} \underbrace{(q_{i_j+1}, \alpha'_j) \vdash^\star_{\mathcal{N}} (q_{i_{j+1}}, \varepsilon)}_{\text{само преходи от } \mathcal{A}}.$$

Понеже имаме, че $\delta(q_{\texttt{start}}, a_{i_j}) = q_{i_j+1}$, то оттук следва, че:

$$\underbrace{(q_{\texttt{start}}, a_{i_j}\alpha'_j) \vdash_{\mathcal{N}} (q_{i_j+1}, \alpha'_j)}_{\text{преход от } \mathcal{A}} \vdash^\star_{\mathcal{A}} (q_{i_{j+1}}, \varepsilon).$$

Заключаваме, че

$$(q_{\texttt{start}}, \alpha_j) \vdash^\star_{\mathcal{A}} (q_{i_{j+1}}, \varepsilon).$$

Понеже $q_{i_{j+1}} \in F$, веднага следва, че $\alpha_j \in \mathcal{L}(\mathcal{A})$. От всичко дотук заключваме, че $\alpha \in (\mathcal{L}(\mathcal{A}))^+$.

За другата посока, нека $\alpha \in (\mathcal{L}(\mathcal{A}))^+$. Това означава, че думата α може да се представи като $\alpha = \alpha_0 \cdot \alpha_1 \cdots \alpha_\ell$, където $\alpha_j \in \mathcal{L}(\mathcal{A})$ и $\alpha_j \neq \varepsilon$, за $j = 0, \ldots, \ell$. Нека за $j = 0, \ldots, \ell$ да положим

$$\alpha_j \stackrel{\text{деф}}{=} a_j \cdot \alpha'_j \text{ и } q_j \stackrel{\text{деф}}{=} \delta(q_{\texttt{start}}, a_j).$$

Оттук получаваме за $j = 0, \ldots, \ell$ следните изчисления:

$$(q_{\texttt{start}}, \alpha_j) \vdash_{\mathcal{A}} (q_j, \alpha'_j) \vdash^\star_{\mathcal{A}} (f_j, \varepsilon), \text{ за някои } f_j \in F.$$

Понеже функцията на преходите на $\mathcal{N}$ разширява функцията на преходите на $\mathcal{A}$, то е ясно е, че имаме също така и следното:

$$(q_{\texttt{start}}, \alpha_j) \vdash_{\mathcal{N}} (q_j, \alpha'_j) \vdash^\star_{\mathcal{N}} (f_j, \varepsilon), \text{ за някои } f_j \in F.$$

За $1 \leq j < \ell$, понеже $\delta(q_{\texttt{start}}, a_j) = q_j$ и $f_{j-1} \in F$, то според конструкцията на недетерминирания краен автомат $\mathcal{N}$,

$$q_j \in \Delta(f_{j-1}, a_j).$$

Оттук следва, че имаме следното изчисление на $\mathcal{N}$ върху α:

$$\begin{array}{ll}
(q_{\texttt{start}}, \alpha_0) \vdash^\star_{\mathcal{N}} (f_0, \varepsilon) & \text{// за някое } f_0 \in F \\
(f_0, \alpha_1) \vdash_{\mathcal{N}} (q_1, \alpha'_1) \vdash^\star_{\mathcal{N}} (f_1, \varepsilon) & \text{// за някое } f_1 \in F \\
(f_1, \alpha_2) \vdash_{\mathcal{N}} (q_2, \alpha'_2) \vdash^\star_{\mathcal{N}} (f_2, \varepsilon) & \text{// за някое } f_2 \in F \\
\ldots & \\
(f_{\ell-1}, \alpha_\ell) \vdash_{\mathcal{N}} (q_\ell, \alpha'_\ell) \vdash^\star_{\mathcal{N}} (f_\ell, \varepsilon). & \text{// за някое } f_\ell \in F
\end{array}$$

Обединявайки всичко това, заключаваме, че $\alpha \in \mathcal{L}(\mathcal{N})$.

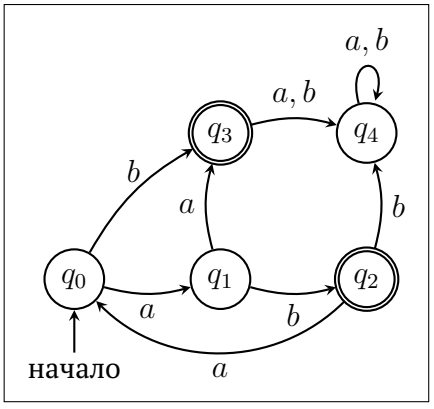
Така доказахме, че $\mathcal{L}(\mathcal{N}) = (\mathcal{L}(\mathcal{A}))^+$. Сега ще построим недетерминиран краен автомат $\mathcal{N}' = \langle \Sigma, Q', q'_{\texttt{start}}, \Delta', F' \rangle$, такъв че:

$$\mathcal{L}(\mathcal{N}') = \mathcal{L}(\mathcal{N}) \cup \{\varepsilon\} = (\mathcal{L}(\mathcal{A}))^+ \cup \{\varepsilon\} = (\mathcal{L}(\mathcal{A}))^\star.$$

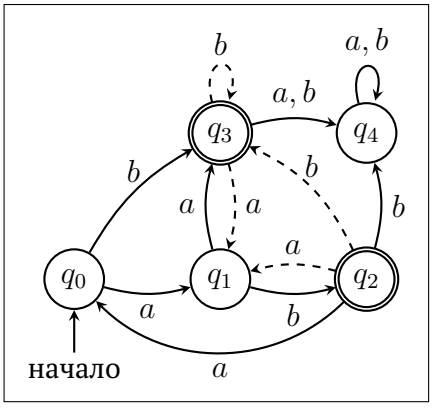
- $Q' \stackrel{\text{деф}}{=} Q \cup \{q'_{\texttt{start}}\}$;
- $F' \stackrel{\text{деф}}{=} F \cup \{q'_{\texttt{start}}\}$;
- $\Delta'(q'_{\texttt{start}}, a) \stackrel{\text{деф}}{=} \Delta(q_{\texttt{start}}, a)$, за всяко $a \in \Sigma$;
- $\Delta'(q, a) \stackrel{\text{деф}}{=} \Delta(q, a)$, за всяко $a \in \Sigma$ и $q \in Q$.

Лесно се съобразява, че $\mathcal{L}(\mathcal{N}') = \mathcal{L}(\mathcal{N}) \cup \{\varepsilon\}$. □

Пример 2.5. Нека да приложим конструкцията за да намерим автомат разпознаващ езика $\mathcal{L}(\mathcal{A})^\star$.



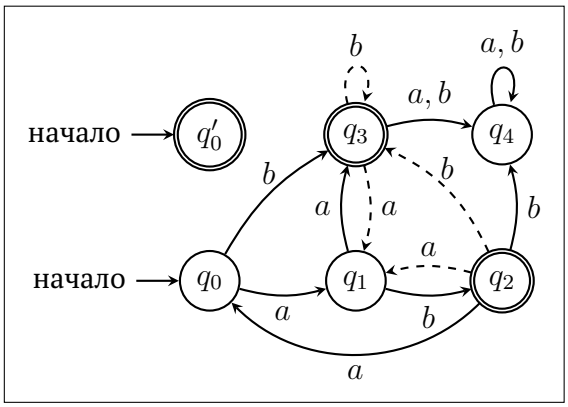
(а) автомат $\mathcal{A}$



(б) $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A})^+$

След като построим автомат за езика $\mathcal{L}(\mathcal{A})^+$, трябва да приложим конструкцията за обединение на автомата за езика $\mathcal{L}(\mathcal{A})^+$ с автомата за езика $\{\varepsilon\}$. Защо трябва да

добавим ново начално състояние q_0'? Да допуснем, че вместо това сме направили q_0 финално. Тогава има опасност да разпознаем повече думи. Например, думата aba би се разпознала от този автомат, но $aba \notin \mathcal{L}(\mathcal{A})^\star$.



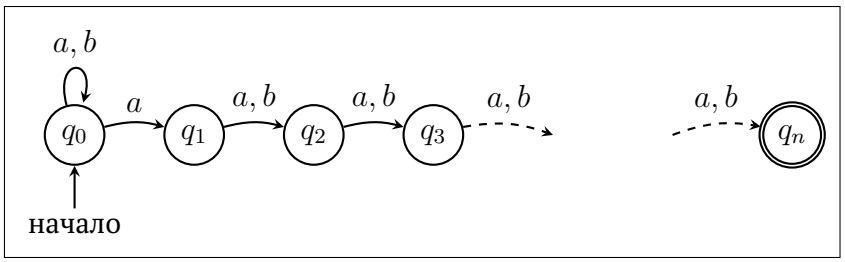
Фигура 2.16: $\mathcal{L}(\mathcal{N}') = \mathcal{L}(\mathcal{N}) \cup \{\varepsilon\} = \mathcal{L}(\mathcal{A})^\star$

2.3.2 Експоненциална експлозия

Тук следваме [KN01, стр. 66] и [ALSU07, стр. 164]. В [Sha08, стр. 80] има друг пример за НКА с n състояния вместо $n+1$, но доказателството изглежда по-сложно.

Твърдение 2.7. Съществува НКА $\mathcal{N}$ с $n+1$ състояния, за който не съществува ДКА $\mathcal{A}$ с по-малко от 2^n състояния.

Упътване. Да разгледаме следния недетерминиран краен автомат $\mathcal{N}$.



Фигура 2.17: Недетерминиран автомат $\mathcal{N}$ за езика $\{a,b\}^\star \cdot \{a\} \cdot \{a,b\}^{n-1}$.

Лесно се съобразява, че за $n > 0$, недетерминирният краен автомат $\mathcal{N}$ на *Фигура 2.17* с $n+1$ на брой състояния разпознава езика

$$L = \{\omega \cdot a \cdot \delta \in \{a,b\}^\star \mid |\delta| = n-1\}.$$

Нека да съобразим, че не е възможно да съществува краен детерминиран автомат $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ разпознаващ същия език L с по-малко от 2^n състояния.

Да допуснем, че $|Q| < 2^n$. От принципа на Дирихле имаме, че съществуват две различни думи α и β с дължина n, за които съществува $q \in Q$ и

Да напомним, че броят на всички думи с дължина n над азбука с k букви е k^n. В нашия случай, $k = 2$.

$$\delta^\star(q_{\texttt{start}}, \alpha) = q = \delta^\star(q_{\texttt{start}}, \beta).$$

Нека първата разлика в тези две думи е на позиция $i < n$, т.е. думите α и β могат да се представят така:

$$\alpha = \omega \cdot \alpha[i] \cdot \rho \text{ и } \beta = \omega \cdot \beta[i] \cdot \delta.$$

Без ограничение на общността, нека $\alpha[i] = a$ и $\beta[i] = b$.

- Ако $i = 0$. Това означава, че $\alpha \in L$, но $\beta \notin L$. Следователно състоянието q е едновременно финално и нефинално. Това е противоречие.

  Тук $\omega = \varepsilon$.

- Ако $i > 0$. Да разгледаме следните думи:

  $$\alpha_0 = \alpha \cdot a^i$$
  $$\beta_0 = \beta \cdot a^i.$$

  Съобразете, че тук $|\omega| = i$ и $|\delta| = n - i - 1$. Не е важно как разширяваме α и β за да получим α_0 и β_0. Тук за да бъдем конкретни сме избрали просто a^i. Важното е това разширение да има дължина i.

  Тогава $\alpha_0 = \omega \cdot a \cdot \delta_0$, където $|\delta_0| = n - 1$. Аналогично, $\beta_0 = \rho \cdot b \cdot \delta_1$, където $|\delta_1| = n - 1$. Така отново получаваме, че $\alpha_0 \in L$, но $\beta_0 \notin L$. И в този случай получаваме противоречие, защото

  $$\delta^\star(q_{\texttt{start}}, \alpha_0) = p = \delta^\star(q_{\texttt{start}}, \beta_0)$$

  и състоянието p трябва да е едновременно финално и нефинално.

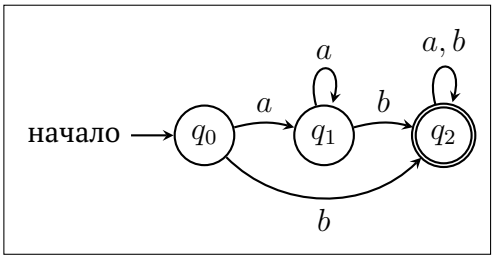
□

2.4 Системи от регулярни изрази

Тази част трябва да се разшири или да се премахне [Sak09, стр. 100].

Лема 2.5 (Ардън)**.** Нека $\mathbf{r}$ и $\mathbf{p}$ са регулярни изрази. Тогава регулярният израз $\mathbf{r}^\star\mathbf{p}$ е най-малкото решение на уравнението $X = \mathbf{r} \cdot X + \mathbf{p}$. Ако $\varepsilon \notin \mathcal{L}(\mathbf{r})$, то това решение е единствено.

Да разгледаме отново автомата от *Фигура 2.8*.



Фигура 2.18: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}(\mathbf{a}^\star\mathbf{b}(\mathbf{a}+\mathbf{b})^\star)$.

На него съответства следната система:

$$\begin{aligned} X_0 &= a \cdot X_1 + b \cdot X_2 + \emptyset \\ X_1 &= a \cdot X_1 + b \cdot X_2 + \emptyset \\ X_2 &= a \cdot X_2 + b \cdot X_2 + \varepsilon. \end{aligned}$$

Чрез прилагане на лемата на Ардън и заместване получаваме следното:

$$\begin{aligned} X_0 &= a \cdot X_1 + b \cdot (a+b)^\star \\ X_1 &= a \cdot X_1 + b \cdot (a+b)^\star \\ X_2 &= (a+b)^\star. \end{aligned}$$

След това,

$$\begin{aligned} X_0 &= a \cdot a^\star \cdot b \cdot (a+b)^\star + b \cdot (a+b)^\star \\ X_1 &= a^\star \cdot b \cdot (a+b)^\star \\ X_2 &= (a+b)^\star. \end{aligned}$$

Заключаваме, че езикът на автомата е

$$(a \cdot a^\star + \varepsilon) \cdot b \cdot (a+b)^\star = a^\star \cdot b \cdot (a+b)^\star.$$

2.5 Един критерий за нерегулярност

Лема 2.6 (Лема за покачването)**.** Нека L да бъде *безкраен* регулярен език. Съществува число $p \geq 1$, зависещо само от L, за което за всяка дума $\alpha \in L, |\alpha| \geq p$ може да бъде записана във вида $\alpha = xyz$ и

На англ. се нарича *Pumping Lemma*. Има подобна лема и за безконтекстни езици, която ще разгледаме по-нататък.

1) $|y| \geq 1$;

2) $|xy| \leq p$;

Обърнете внимание, че $0 \in \mathbb{N}$ и $xy^0z = xz$

3) $(\forall i \in \mathbb{N})[xy^iz \in L]$.

Упътване. Понеже L е регулярен, то L е и автоматен език. Нека

Тази лема я има на практика във всеки добър учебник по този предмет. Например, [PL98, стр. 88], [Sip12, стр. 77], [HU79, стр. 55].

$$\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$$

е краен детерминиран автомат, за който $L = \mathcal{L}(\mathcal{A})$. Да положим $p = |Q|$ и нека $\alpha = a_1a_2 \cdots a_k$ е дума, за която $k \geq p$. Да разгледаме първите p стъпки от изпълнението на α върху $\mathcal{A}$:

$$q_{\texttt{start}} \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \ldots \xrightarrow{a_p} q_p.$$

Тъй като $|Q| = p$, а по този път участват $p+1$ състояния $q_0, q_1, \ldots, q_p$, то съществуват числа i, j, за които $0 \leq i < j \leq p$ и $q_i = q_j$. Нека разделим думата α на три части по следния начин:

$$\underbrace{a_1 \cdots a_i}_{x} \quad \underbrace{a_{i+1} \cdots a_j}_{y} \quad \underbrace{a_{j+1} \cdots a_k}_{z}.$$

Ясно е, че $|y| \geq 1$ и $|xy| = j \leq p$. Да разгледаме случая за $i = 0$. Думата $xy^0z = xz \in L$, защото имаме следното изчисление:

$$q_{\texttt{start}} \underbrace{\xrightarrow{a_1} q_1 \cdots \xrightarrow{a_i}}_{x} q_i \underbrace{\xrightarrow{a_{j+1}} q_{j+1} \cdots \xrightarrow{a_k}}_{z} q_k \in F,$$

защото $q_i = q_j$. Да разгледаме и случая $i = 2$. Тогава думата $xy^2z \in L$, защото имаме следното изчисление:

$$q_{\texttt{start}} \underbrace{\xrightarrow{a_1} q_1 \cdots \xrightarrow{a_i}}_{x} q_i \underbrace{\xrightarrow{a_{i+1}} q_{i+1} \cdots \xrightarrow{a_j}}_{y} q_j \underbrace{\xrightarrow{a_{i+1}} q_{i+1} \cdots \xrightarrow{a_j}}_{y} q_j \underbrace{\xrightarrow{a_{j+1}} \cdots \xrightarrow{a_k}}_{z} q_k \in F.$$

Вече лесно можем да съобразим, че за всяко естествено число i, е изпълнено $xy^iz \in \mathcal{L}(\mathcal{A})$. □

✍ Докажете!

Практически е по-полезно да разглеждаме следната еквивалентна формулировка на лемата за покачването.

Подобно е изложението и в [Koz97, стр. 70].

Следствие 2.1 (Контрапозиция на лемата за покачването)**.** Нека L е произволен език. Нека също така е изпълнено, че:

(∀) за *всяко* естествено число $p \geq 1$,

(∃) можем да намерим дума $\alpha \in L$, $|\alpha| \geq p$, такава че

(∀) за *всяко* разбиване на думата на три части, $\alpha = xyz$, със свойствата $|xy| \leq p$ и $|y| \geq 1$,

(∃) можем да посочим $i \in \mathbb{N}$, за което е изпълнено, че $xy^iz \notin L$.

Тогава L **не** е регулярен език.

Доказателство. Да означим с $P_{\text{reg}}(L)$ следната формула:

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

Понеже имаме, че $p \to q \equiv \neg p \vee q$, то горната формула може да се запише и така:

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

Така условието на Лемата за покачването представлява твърдението:

„Ако L е регулярен език, то е изпълнено $P_{reg}(L)$."

Контрапозиция на твърдението $P \to Q$ е твърдението $\neg Q \to \neg P$

Лемата може да се запише по следния еквивалентен начин:

„Ако $P_{reg}(L)$ не е изпълнено, то L не е регулярен език."

Или еквивалентно,

„Ако $\neg P_{reg}(L)$ е изпълнено, то L не е регулярен език."

Използваме, че $\neg\exists\forall\exists(\dots) \equiv \forall\exists\forall\neg(\dots)$

Сега $\neg P_{\text{reg}}(L)$ преставлява формулата

$$(\forall p \geq 1)(\exists \alpha \in L)[\, |\alpha| \geq p \;\wedge\; (\forall x, y, z \in \Sigma^\star)[\, \alpha \neq xyz \vee |y| \ngeq 1 \vee |xy| \nleq p \vee (\exists i \in \mathbb{N})[xy^iz \notin L]\,]\,].$$

Използваме, че

$$\frac{\neg P \vee \neg Q \vee \neg S \vee R}{\dfrac{\neg(P \wedge Q \wedge S) \vee R}{(P \wedge Q \wedge S) \to R}}$$

Горната формула е еквивалентна на:

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

Това означава, че ако

(∀) вземем произволна константа $p \geq 1$,

(∃) за нея намерим конкретна дума $\alpha \in L$, такава че $|\alpha| \geq p$ и

(∀) докажем, че за всяко нейно разбиване на три части x, y, z, със свойствата $|y| \geq 1$ и $|xy| \leq p$,

(∃) можем да посочим естествено число i, за което $xy^iz \notin L$,

то можем да заключим, че езикът L не е регулярен. □

Пример 2.6. Да видим защо езикът $L = \{a^nb^n \mid n \in \mathbb{N}\}$ не е регулярен. За да направим това като използваме контрапозицията на лемата за покачването, то трябва да докажем, че $\neg P_{\text{reg}}(L)$ е изпълнено. Както обяснихме по-горе, доказателството следва стъпките:

Това е важен пример. По-късно ще видим, че този език е безконтекстен.

(∀) Разглеждаме произволно число $p \geq 1$.

(∀) Нямаме власт над избора на числото p.

(∃) Избираме дума $\alpha \in L$, за която $|\alpha| \geq p$. Имаме свободата да изберем каквато дума α си харесаме, стига тя да принадлежи на L и да има дължина поне p. Щом имаме тази свобода, нека да изберем думата $\alpha = a^pb^p \in L$. Очевидно е, че $|\alpha| \geq p$.

(∃) Няма общо правило, което да ни казва как избираме думата α. Трябва сами да се досетим. Обърнете внимание, че думата α зависи от константата p.

(∀) Разглеждаме произволно разбиване на α на три части, $\alpha = xyz$, за които изискваме свойствата $|xy| \leq p$ и $|y| \geq 1$.

(∀) Не знаем нищо друго за x, y и z освен тези две свойства.

(∃) Ще намерим $i \in \mathbb{N}$, за което $xy^iz \notin L$. Понеже $|xy| \leq p$, то $y = a^k$, за $1 \leq k \leq p$. Тогава ако вземем $i = 0$, получаваме $xy^0z = a^{p-k}b^p$. Ясно е, че $xz \notin L$, защото $p - k < p$.

(∃) Изборът на i може да зависи разбиването x, y, z. В конкретния пример не зависи.

Тогава от *Следствие 2.1* следва, че L не е регулярен език.

Забележка. Много често студентите правят следното разсъждение:

✍ Съобразете сами!

$$L \text{ е регулярен } \& \; L' \subseteq L \implies L' \text{ е регулярен.}$$

Съобразете, че в общия случай това твърдение е *невярно*. За да видим това, достатъчно е да посочим регулярен език L, който има като подмножество нерегулярен език L'. Също лесно се вижда, че твърдението

$$L \text{ е регулярен } \& \; L \subseteq L' \implies L' \text{ е регулярен}$$

е невярно.

Поради подобни съображения, следните твърдения също са *неверни*:

$$L \text{ не е регулярен } \& \; L' \subseteq L \implies L' \text{ не е регулярен}$$
$$L' \text{ не е регулярен } \& \; L' \subseteq L \implies L \text{ не е регулярен.}$$

Следствия

Твърдение 2.8. Езикът на автомата $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ е *непразен* точно тогава, когато съдържа дума α, за която $|\alpha| < |Q|$.

Доказателство. Ще разгледаме двете посоки на твърдението.

($\Rightarrow$) Нека L е непразен език и нека $m = \min\{|\alpha| \mid \alpha \in L\}$. Ще докажем, че $m < |Q|$. За целта, да допуснем, че $m \geq |Q|$ и да изберем $\alpha \in L$, за която $|\alpha| = m$. Според доказателството на *Лема 2.6*, съществува разбиване $xyz = \alpha$, такова че $xz \in L$. При положение, че $|y| \geq 1$, то $|xz| < m$, което е противоречие с минималността на думата α. Заключаваме, че нашето допускане е грешно. Тогава $m < |Q|$, откъдето следва, че съществува дума $\alpha \in L$ с $|\alpha| < |Q|$.

($\Leftarrow$) Тази посока е тривиална. Ако L съдържа дума α, за която $|\alpha| < |Q|$, то е очевидно, че L е непразен език.

□

✍ Обяснете!

Следствие 2.2. Съществува алгоритъм, който проверява дали даден регулярен език е празен или не.

За *Следствие 2.3*, съобразете, че $L_1 = L_2$ точно тогава, когато $(L_1 \setminus L_2) \cup (L_2 \setminus L_1) = \emptyset$.

Следствие 2.3. Съществува алгоритъм, който определя дали два автомата $\mathcal{A}_1$ и $\mathcal{A}_2$ разпознават един и същ език.

Твърдение 2.9. Регулярният език L, разпознаван от КДА $\mathcal{A}$, е *безкраен* точно тогава, когато съдържа дума $\alpha, |Q| \leq |\alpha| < 2|Q|$.

Доказателство. Да разгледаме двете посоки на твърдението.

($\Leftarrow$) Нека L е регулярен език, за който съществува дума α, такава че $|Q| \leq |\alpha| < 2|Q|$. Тогава от *Лема 2.6* следва, че съществува разбиване $\alpha = xyz$ със свойството, че за всяко $i \in \mathbb{N}$, $xy^iz \in L$. Следователно, L е безкраен, защото $|y| \geq 1$.

($\Rightarrow$) Нека L е безкраен език и да вземем *най-късата* дума $\alpha \in L$, за която $|\alpha| \geq 2|Q|$. Понеже L е безкраен, знаем, че такава дума съществува. Тогава отново по *Лема 2.6*, имаме следното разбиване на α:

$$\alpha = xyz,\ |xy| \leq |Q|,\ 1 \leq |y|,\ xz \in L.$$

Но понеже $|xyz| \geq 2|Q|$, а $1 \leq |y| \leq |Q|$, то $|xyz| > |xz| \geq |Q|$ и понеже избрахме $\alpha = xyz$ да бъде най-късата дума с дължина поне $2|Q|$, заключаваме, че $|Q| \leq |xz| < 2|Q|$ и $xz \in L$.

□

✍ Обяснете!

Следствие 2.4. Съществува алгоритъм, който проверява дали даден регулярен език е безкраен.

✍ Обяснете!

Следствие 2.5. Съществува алгоритъм, който проверява дали симетричната разлика на два регулярни езика е крайна.

Примерни задачи

Задача 2.9. Докажете, че езикът $L = \{a^m b^n \mid m, n \in \mathbb{N} \,\&\, m < n\}$ не е регулярен.

Доказателство. Доказателството следва стъпките:

На стъпка от вида $(\forall)$ нямаме власт над това как избираме съответния елемент. На стъпка от вида $(\exists)$ имаме тази власт. Тогава трябва да посочим конкретен елемент.

$(\forall)$ Разглеждаме произволно число $p \geq 1$.

$(\exists)$ Избираме дума $\alpha \in L$, за която $|\alpha| \geq p$. Имаме свободата да изберем каквато дума α си харесаме, стига тя да принадлежи на L и да има дължина поне p. Щом имаме тази свобода, нека да изберем думата $\alpha = a^p b^{p+1} \in L$. Очевидно е, че $|\alpha| \geq p$.

$(\forall)$ Разглеждаме произволно разбиване на α на три части, $\alpha = xyz$, за които изискваме свойствата $|xy| \leq p$ и $|y| \geq 1$.

$(\exists)$ Ще намерим конкретно $i \in \mathbb{N}$, за което $xy^i z \notin L$. Понеже $|xy| \leq p$, то $y = a^k$, за $1 \leq k \leq p$. Тогава ако вземем $i = 2$, получаваме

Тук изборът на i не зависи от изборите, които сме направили на предишните стъпки.

$$xy^2z = a^{p-k}a^{2k}b^{p+1} = a^{p+k}b^{p+1}.$$

Ясно е, че $xy^2z \notin L$, защото $p + k \geq p + 1$.

Тогава от контрапозицията на лемата за покачването следва, че L не е регулярен език. □

Задача 2.10. Докажете, че езикът $L = \{a^n \mid n \text{ е просто число}\}$ не е регулярен.

Доказателство. Доказателството следва стъпките:

$(\forall)$ Разглеждаме произволно число $p \geq 1$.

$(\exists)$ Избираме дума $w \in L$, за която $|w| \geq p$. Можем да изберем каквото w си харесаме, стига то да принадлежи на L и да има дължина поне p. Нека да изберем една конкретна дума $w \in L$, такава че $|w| > p + 1$. Знаем, че такава дума съществува, защото L е безкраен език. По-долу ще видим защо този избор е важен за нашите разсъждения.

$(\forall)$ Разглеждаме произволно разбиване на w на три части, $w = xyz$, за които изискваме свойствата $|xy| \leq p$ и $|y| \geq 1$.

Обърнете внимание, че тук е по-интересно. Изборът на i зависи от предишната стъпка, на която сме разбили думата w на три части.

$(\exists)$ Ще намерим конкретно i, за което $xy^i z \notin L$, т.е. ще намерим i, за което $|xy^i z| = |xz| + i \cdot |y|$ е *съставно число*. Понеже $|xy| \leq p$ и $|xyz| > p + 1$, то $|z| > 1$. Да изберем $i = |xz| > 1$. Тогава:

$$|xy^i z| = |xz| + i.|y| = |xz| + |xz|.|y| = (1 + |y|)|xz|$$

е съставно число, следователно $xy^i z \notin L$.

Изискваме $|w| > p + 1$, защото искаме да гарантираме, че $|xz| > 1$.

Тогава от контрапозицията на лемата за покачването следва, че L не е регулярен език. □

Задача 2.11. Докажете, че езикът $L = \{a^{n^2} \mid n \in \mathbb{N}\}$ не е регулярен.

Доказателство. В тази задача ще използваме следното свойство:

$$n \text{ не е точен квадрат } \Leftrightarrow (\exists p \in \mathbb{N})[p^2 < n < (p+1)^2].$$

Доказателството следва стъпките:

$(\forall)$ Разглеждаме произволно число $p \geq 1$.

$(\exists)$ Избираме достатъчно дълга дума, която принадлежи на езика L. За да бъдем конкретни, нека $w = a^{p^2}$.

$(\forall)$ Разглеждаме произволно разбиване на w на три части, $w = xyz$, като $|xy| \leq p$ и $|y| \geq 1$.

($\exists$) Ще намерим конкретно i, за което $xy^iz \notin L$. В нашия случай това означава, че $|xz| + i \cdot |y|$ не е точен квадрат. Тогава за $i = 2$,

$$p^2 = |xyz| < |xy^2z| = |xyz| + |y| \leq p^2 + p < p^2 + 2p + 1 = (p+1)^2.$$

Получаваме, че $p^2 < |xy^2z| < (p+1)^2$, откъдето следва, че $|xy^2z|$ не е точен квадрат. Следователно, $xy^2z \notin L$.

Тогава от контрапозицията на лемата за покачването следва, че L не е регулярен език. □

Задача 2.12. Докажете, че езикът $L = \{a^{n!} \mid n \in \mathbb{N}\}$ не е регулярен.

Доказателство. Доказателството следва стъпките:

($\forall$) Разглеждаме произволно число $p \geq 1$.

($\exists$) Избираме достатъчно дълга дума, която принадлежи на езика L. За да бъдем конкретни, нека $\omega = a^{(p+1)!}$.

($\forall$) Разглеждаме произволно разбиване на ω на три части, $\omega = xyz$, като $|xy| \leq p$ и $|y| \geq 1$. Да обърнем внимание, че $1 \leq |y| \leq p$.

($\exists$) Ще намерим конкретно i, за което $xy^iz \notin L$. Това означава да съществува n, за което

$$n! < |xy^iz| < (n+1)!$$

Да разгледаме $i = 2$. Тогава:

$$\begin{aligned}(p+1)! &< |xy^2z| \\ &= (p+1)! + |y| \\ &\leq (p+1)! + p \\ &< (p+1)! + (p+1)!(p+1) \\ &= (p+2)!\end{aligned}$$

Възможно е да вземем $w = a^{(p+2)!}$. Тогава възможно ли е $xy^0z \notin L$? Понеже $|xyz| = (p+2)!$, това означава, че би трябвало $|xz| = k!$, за някое $k \leq p+1$. Тогава

$$\begin{aligned}|y| &= |xyz| - |xz| \\ &= (p+2)! - k! \\ &\geq (p+2)! - (p+1)! \\ &= (p+1).(p+1)! \\ &> p.\end{aligned}$$

Достигнахме до противоречие с условието, че $|y| \leq p$.

Тогава от контрапозицията на лемата за покачването следва, че L не е регулярен език. □

Задача 2.13. Докажете, че езикът $L = \{\alpha\beta \in \{a,b\}^\star \mid |\alpha| = |\beta| \;\&\; \alpha \neq \beta\}$ не е регулярен.

Упътване. Да допуснем, че L е регулярен. Тогава езикът $\overline{L} = \{a,b\}^\star \setminus L$ също е регулярен. Ясно е, че

$$\overline{L} = \{\alpha\beta \in \{a,b\}^\star \mid \alpha = \beta\} \cup \{\omega \in \{a,b\}^\star \mid |\omega| \text{ е нечетно число}\}.$$

Тогава езикът $L_1 = \overline{L} \cap \{\omega \in \{a,b\}^\star \mid |\omega| \text{ е четно число}\}$ също е регулярен. Ясно е, че $L_1 = \{\alpha\beta \in \{a,b\}^\star \mid \alpha = \beta\}$. Сега можем да разгледаме регулярния език

$$L_2 = L_1 \cap \mathcal{L}(\mathbf{a^\star ba^\star b}) = \{a^n b a^n b \mid n \in \mathbb{N}\}.$$

За него вече лесно можем да приложим Лемата за покачването и да получим, че L_2 не е регулярен. Така достигаме до противоречие с допускането, че L е регулярен. □

Пример, за който лемата не е приложима

Да напомним, че условието на Лемата за покачването представлява твърдението:

„Ако L е регулярен език, то е изпълнено $P_{reg}(L)$."

Сега ще видим, че можем да посочим език L, който не е регулярен, но въпреки това условието $P_{\text{reg}}(L)$ е изпълнено. Това означава, че нямаме обратната посока на горната импликация и може да срещнем примери за езици, които макар и нерегулярни, не можем да докажем тяхната нерегулярност с помощта на контрапозицията на лемата за покачването. По-късно ще видим един пълен критерий за проверка за регулярност на език.

Пример 2.7. Езикът

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

не е регулярен, но условието $P_{\text{reg}}(L)$ е изпълнено.

Упътване. Ако допуснем, че L е регулярен, то тогава ще следва, че

$$L_1 = L \cap \mathcal{L}(\mathbf{ca^\star b^\star}) = \{ca^n b^n \mid n \in \mathbb{N}\}$$

е регулярен, но с контрапозицията на лемата за покачването лесно се вижда, че L_1 не е. Сега да проверим, че $P_{\text{reg}}(L)$ е изпълнено.

(∃) Нека изберем $p = 2$.

(∀) Сега трябва да разгледаме всички думи $\alpha \in L$, $|\alpha| \geq 2$.

(∃) Нека разбием думата α на три части по следния начин:

$$x = \varepsilon,\ y = \alpha[0],\ z = \alpha[1:].$$

(∀) Съобразете, че за всяко $i \in \mathbb{N}$ имаме, че $xy^i z \in L$.

□

За да покажем, че $P_{\text{reg}}(L)$ е изпълнено, трябва да следваме стъпките:

(∃) Избираме конкретно число $p \geq 1$.

(∀) Разглеждаме произволна дума $\alpha \in L$ и $|\alpha| \geq p$.

(∃) Посочваме конкретно разбиване на думата α като $\alpha = xyz$ със свойството $|xy| \leq p$ и $|y| \geq 1$.

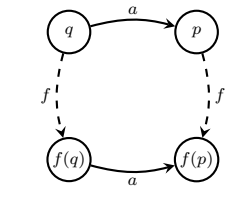
(∀) За всяко i трябва да покажем, че $xy^i z \in L$.

2.6 Изоморфни автомати

[Koz97, стр. 89]

Нека са дадени автоматите $\mathcal{A}_1 = (\Sigma, Q_1, q'_{\texttt{start}}, \delta_1, F_1)$ и $\mathcal{A}_2 = (\Sigma, Q_2, q''_{\texttt{start}}, \delta_2, F_2)$. Казваме, че $\mathcal{A}_1$ и $\mathcal{A}_2$ са **изоморфни**, което означаваме с $\mathcal{A}_1 \cong \mathcal{A}_2$, ако съществува *биективна* функция $f : Q_1 \to Q_2$, за която:

Условие (3) може да се представи графично така:



(1) $f(q'_{\texttt{start}}) = q''_{\texttt{start}}$;

(2) $q \in F_1 \Leftrightarrow f(q) \in F_2$;

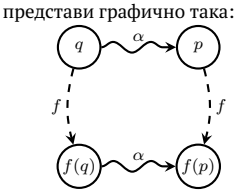
(3) $\delta_1(q, a) = p \Leftrightarrow \delta_2(f(q), a) = f(p)$.

Ще казваме, че f задава изоморфизъм на $\mathcal{A}_1$ върху $\mathcal{A}_2$ и ще означаваме $\mathcal{A}_1 \cong_f \mathcal{A}_2$ или $f : \mathcal{A}_1 \cong \mathcal{A}_2$.

Твърдение 2.10. Нека $\mathcal{A}_1 \cong_f \mathcal{A}_2$. Тогава

$$\delta_1^\star(q, \alpha) = p \Leftrightarrow \delta_2^\star(f(q), \alpha) = f(p). \tag{2.7}$$

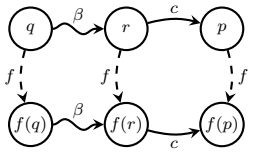
Свойство (2.7) може да се представи графично така:



Доказателство. Както винаги, ще докажем *Свойство (2.7)* с индукция по дължината на думата α.

- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Ясно е, че *Свойство (2.7)* е изпълнено за ε защото $\delta_1^\star(q, \varepsilon) = q$ и съответно $\delta_2^\star(f(q), \varepsilon) = f(q)$ за произволно състояние $q \in Q_1$.
- Да приемем, че *Свойство (2.7)* е изпълнено за думи с дължина n.
- Да разгледаме произволна дума α с дължина $n+1$, т.е. $\alpha = \beta c$ и $|\beta| = n$. Тогава:

Индукционната стъпка може да се представи така:



$$\delta_1^\star(q, \beta c) = p \Leftrightarrow \delta_1(\underbrace{\delta_1^\star(q, \beta)}_{r}, c) = p.$$

Понеже f е изоморфизъм, то

$$\delta_1(r, c) = p \Leftrightarrow \delta_2(f(r), c) = f(p).$$

От **(И.П.)** за *Свойство (2.7)* имаме, че:

$$\delta_1^\star(q, \beta) = r \Leftrightarrow \delta_2^\star(f(q), \beta) = f(r).$$

Комбинираме тези две еквивалентности и получаваме следното:

$$\begin{aligned}
\delta_1^\star(q, \beta c) = p &\Leftrightarrow \delta_1(\underbrace{\delta_1^\star(q, \beta)}_{r}, c) = p \\
&\Leftrightarrow \delta_1(r, c) = p \\
&\Leftrightarrow \delta_2(f(r), c) = f(p) \\
&\Leftrightarrow \delta_2(\delta_2^\star(f(q), \beta), c) = f(p) \\
&\Leftrightarrow \delta^\star(f(q), \beta c) = f(p).
\end{aligned}$$

□

Твърдение 2.11. Ако $\mathcal{A}_1 \cong \mathcal{A}_2$, то $\mathcal{L}(\mathcal{A}_1) = \mathcal{L}(\mathcal{A}_2)$.

Упътване. Нека $\mathcal{A}_1 \cong_f \mathcal{A}_2$. Тогава имаме следните еквивалентности:

Лесно можем да съобразим, че в общия случай нямаме обратната посока на това твърдение.

$$
\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}_1) &\Leftrightarrow \delta_1^\star(q'_{\mathtt{start}}, \alpha) = p \in F_1 && \text{// деф. на } \mathcal{L}(\mathcal{A}_1)\\
&\Leftrightarrow \delta_2^\star(f(q'_{\mathtt{start}}), \alpha)) = f(p) \in F_2 && \text{// } \textit{Свойство (2.7)}\\
&\Leftrightarrow \delta_2^\star(q''_{\mathtt{start}}, \alpha) \in F_2 && \text{// } f(q'_{\mathtt{start}}) \overset{\text{деф}}{=} q''_{\mathtt{start}}\\
&\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}_2). && \text{// деф. на } \mathcal{L}(\mathcal{A}_2)
\end{aligned}
$$

□

2.7 Автомат на Бжозовски

Имаме следната операция за произволна буква a,

$$a^{-1}(L) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid a \cdot \omega \in L\}.$$

Бжозовски [Brz64] описва алгоритъм за строене на автомат по регулярен израз [Sak09, стр. 112].

Да видим как се държи тази операция върху регулярните езици. За целта следваме дефиницията на регулярните езици.

Задача 2.14. Докажете, че за произволна буква a и език L са изпълнени равенствата:

(1) $a^{-1}(\emptyset) = \emptyset$;

(2) $a^{-1}(\{\varepsilon\}) = \emptyset$;

(3) $a^{-1}(\{b\}) = \begin{cases} \{\varepsilon\}, & \text{ако } a = b \\ \emptyset, & \text{ако } a \neq b \end{cases}$

(4) $a^{-1}(L_1 \cup L_2) = a^{-1}(L_1) \cup a^{-1}(L_2)$;

(5) $a^{-1}(L_1 \cdot L_2) = \begin{cases} a^{-1}(L_1) \cdot L_2, & \text{ако } \varepsilon \notin L_1 \\ a^{-1}(L_1) \cdot L_2 \cup a^{-1}(L_2), & \text{ако } \varepsilon \in L_1 \end{cases}$

(6) $a^{-1}(L^\star) = a^{-1}(L) \cdot L^\star$.

Аналогично, за произволна дума α, нека

$$\alpha^{-1}(L) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \alpha\omega \in L\}.$$

Твърдение 2.12. За всеки две думи α и β е изпълнено, че:

$$(\alpha \cdot \beta)^{-1}(L) = \beta^{-1}(\alpha^{-1}(L)).$$

Доказателство. За прозиволна дума γ имаме следните еквивалентности:

$$\begin{aligned} \gamma \in \beta^{-1}(\alpha^{-1}(L)) &\Leftrightarrow \beta\gamma \in \alpha^{-1}(L) \\ &\Leftrightarrow \alpha\beta\gamma \in L \\ &\Leftrightarrow \gamma \in (\alpha\beta)^{-1}(L). \end{aligned}$$

□

Задача 2.15. Докажете, че ако L е регулярен език, то $\alpha^{-1}(L)$ е регулярен език.

Задача 2.16. Докажете, че $L = \{\omega \in \Sigma^\star \mid \varepsilon \in \omega^{-1}(L)\}$.

Нека е даден езикът L. Ще покажем конструкция на детерминиран автомат $\mathcal{B} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$, който разпознава L. Ако L е регулярен, то $\mathcal{B}$ ще бъде детерминиран краен автомат, но ако L не е регулярен, то $\mathcal{B}$ ще бъде детерминиран *безкраен* автомат. Конструкцията на автомата $\mathcal{B}$ е следната:

- Състоянията Q ще бъдат от вида $\hat{M}$, за $M \subseteq \Sigma^\star$, където:

$$Q \stackrel{\text{деф}}{=} \{\hat{M} \mid (\exists \alpha \in \Sigma^\star)[M = \alpha^{-1}(L)]\}.$$

- $q_{\texttt{start}} \stackrel{\text{деф}}{=} \hat{L}$.
- За произволни състояния $\hat{M}$ и $\hat{N}$ и буква a,

$$\delta(\hat{M}, a) = \hat{N} \stackrel{\text{деф}}{\Leftrightarrow} N = a^{-1}(M).$$

- $F \stackrel{\text{деф}}{=} \{\hat{M} \in Q \mid \varepsilon \in M\}$.

Твърдение 2.13. Нека $\mathcal{B}$ е автоматът на Бжозовски за езика L. За всяка дума α и всяко състояние $\hat{M} \in Q^{\mathcal{B}}$ е изпълнена еквивалентността:

$$N = \alpha^{-1}(M) \Leftrightarrow \delta^\star(\hat{M}, \alpha) = \hat{N}. \tag{2.8}$$

Упътване. Индукция по дължината на думата α, като използвате, че

$$(\alpha b)^{-1}(M) = b^{-1}(\alpha^{-1}(M)).$$

✍ Опитайте се да докажете това твърдение сами!

□

Доказателство.

- Твърдението очевидно е изпълнено за $\alpha = \varepsilon$.
- Да примем, че за думи α с дължина n е изпълнено Свойство (2.8).
- Сега да разгледаме дума α с дължина $n+1$, т.е. $\alpha = \beta a$ и $|\beta| = n$. Тогава

$$\begin{aligned}
\delta^\star(\hat{M}, \beta a) &= \delta(\delta^\star(\hat{M}, \beta), a) && \text{// от деф.} \\
&= \delta(\hat{K}, a) && \text{// нека } \hat{K} \stackrel{\text{деф}}{=} \delta^\star(\hat{M}, \beta) \\
&= \hat{N}. && \text{// } N = a^{-1}(K)
\end{aligned}$$

Свойство (2.8) приложено за β ни казва, че

$$\delta^\star(\hat{M}, \beta) = \hat{K} \Leftrightarrow \beta^{-1}(M) = K.$$

От дефиницията на δ имаме, че

$$\delta(\hat{K}, a) = \hat{N} \Leftrightarrow N = a^{-1}(K).$$

Накрая заключаваме, че

$$\delta^\star(\hat{M}, \beta a) = \hat{N} \Leftrightarrow N = a^{-1}(\beta^{-1}(M)) = \alpha^{-1}(M).$$

□

Твърдение 2.14. За даден език L, нека $\mathcal{B}$ е детерминираният автомат построен по метода на Бжозовски. Тогава $L = \mathcal{L}(\mathcal{B})$.

Тук е важно да отбележим, че все още не знаем дали $\mathcal{B}$ има крайно много състояния. В *Пример 2.8* ще видим един детерминиран безкраен автомат за език, който не е регулярен.

Упътване. Съобразете, че имаме следните еквивалентности:

$$\begin{aligned}
\alpha \in L &\Leftrightarrow \varepsilon \in \alpha^{-1}(L) && \text{// от } \textit{Задача 2.16}\\
&\Leftrightarrow \varepsilon \in M \;\&\; \hat{M} = \delta^\star(\hat{L}, \alpha) && \text{// } M \overset{\text{деф}}{=} \alpha^{-1}(L) \text{ от } \textit{Твърдение 2.13}\\
&\Leftrightarrow \delta^\star(q_{\texttt{start}}, \alpha) \in F && \text{// } q_{\texttt{start}} \overset{\text{деф}}{=} \hat{L} \text{ и } \varepsilon \in \alpha^{-1}(L)\\
&\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{B}).
\end{aligned}$$

□

Примерни задачи

Тук $\boldsymbol{r}^+ \stackrel{\text{деф}}{=} \boldsymbol{r} \cdot \boldsymbol{r}^\star$.

Задача 2.17. Постройте автомат $\mathcal{B}$ по метода на Бжозовски за регулярния език

$$L = \mathcal{L}(((\boldsymbol{a} + \boldsymbol{b})^+ \cdot \boldsymbol{a})^\star).$$

Решение.

- Ясно е, че ще започнем от началното състояние $\hat{L}$. Удобно е да имаме предвид следното представяне

$$\begin{aligned} L &= (\{a,b\}^+ \cdot \{a\})^\star \\ &= \{\varepsilon\} \cup (\{a,b\}^+ \cdot \{a\})^+ \\ &= \{\varepsilon\} \cup \{a,b\}^+ aL \\ &= \{\varepsilon\} \cup \{a,b\} \cdot \{a,b\}^\star aL \\ &= \{\varepsilon\} \cup a\{a,b\}^\star aL \cup b\{a,b\}^\star aL \end{aligned}$$

- Сега като имаме това представяне на L, лесно се съобразява, че

$$a^{-1}(L) = b^{-1}(L) = \{a,b\}^\star aL.$$

Нека положим $M \stackrel{\text{деф}}{=} \{a,b\}^\star aL$. Лесно се съобразява, че $M \neq L$, защото $\varepsilon \in L$, но $\varepsilon \notin M$. Това означава, че имаме ново състояние $\hat{M}$ и

$$\begin{aligned} \delta(\hat{L}, a) &\stackrel{\text{деф}}{=} \hat{M} \quad \text{// защото } a^{-1}(L) = M \\ \delta(\hat{L}, b) &\stackrel{\text{деф}}{=} \hat{M}. \quad \text{// защото } b^{-1}(L) = M \end{aligned}$$

- За следващата стъпка е удобно да представим езика M по следния начин:

$$\begin{aligned} M &= \{a,b\}^\star aL \\ &= aL \cup \{a,b\}^+ aL \\ &= aL \cup \{a,b\} \cdot \{a,b\}^\star aL \\ &= aL \cup \{a,b\} \cdot M \\ &= aL \cup aM \cup bM \end{aligned}$$

От това представяне на M веднага се съобразява, че

$$\begin{aligned} a^{-1}(M) &= L \cup M \\ b^{-1}(M) &= M. \end{aligned}$$

Нека да положим $N \stackrel{\text{деф}}{=} L \cup M$. Имаме, че $N \neq L$, защото $a \in N$, но $a \notin L$. Освен това, $N \neq M$, защото $\varepsilon \in N$, но $\varepsilon \notin M$. Това означава, че имаме ново състояние $\hat{N}$ и тогава

$$\begin{aligned} \delta(\hat{M}, a) &\stackrel{\text{деф}}{=} \hat{N} \quad \text{// защото } a^{-1}(M) = N \\ \delta(\hat{M}, b) &\stackrel{\text{деф}}{=} \hat{M} \quad \text{// защото } b^{-1}(M) = M. \end{aligned}$$

- Да разгледаме следното представяне:

$$\begin{aligned} N &= L \cup M \\ &= \{\varepsilon\} \cup aM \cup bM \cup M \\ &= \{\varepsilon\} \cup aM \cup bM \cup aL \\ &= \{\varepsilon\} \cup aN \cup bM. \end{aligned}$$

Очевидно е, че $\boldsymbol{r}^\star = \boldsymbol{\varepsilon} + \boldsymbol{r}^+$.

Веднага се вижда, че $\hat{L}$ ще бъде и финално състояние, защото $\varepsilon \in L$.

Веднага можем да съобразим, че

$$\begin{aligned} a^{-1}(N) &= N \\ b^{-1}(N) &= M. \end{aligned}$$

Сега полагаме:

$$\begin{aligned} \delta(\hat{N}, a) &\stackrel{\text{деф}}{=} \hat{N} \quad \text{// защото } a^{-1}(N) = N \\ \delta(\hat{N}, b) &\stackrel{\text{деф}}{=} \hat{M} \quad \text{// защото } b^{-1}(N) = M. \end{aligned}$$

- Нямаме повече нови състояния. Следователно,

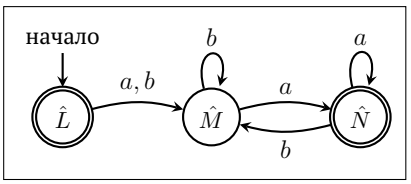
$$Q \stackrel{\text{деф}}{=} \{\hat{L}, \hat{M}, \hat{N}\}.$$

Веднага се вижда, че $\hat{M}$ няма да бъде финално състояние.

- Понеже $\varepsilon \in L$ и $\varepsilon \in N$ е ясно, че

$$F = \{\hat{L}, \hat{N}\}.$$

Сега вече сме готови да нарисуваме картинка на автомата.



Обърнете внимание, че $L \subset N$, но $\hat{L}$ и $\hat{N}$ са различни състояния.

Фигура 2.19: Автомат за езика L по метода на Бжозовски.

□

Задача 2.18. Постройте автомат $\mathcal{B}$ по метода на Бжозовски, който рапознава регулярния език

$$L = \mathcal{L}(\mathbf{a} \cdot (\mathbf{a} + \mathbf{b})^\star \cdot \mathbf{b}).$$

Решение.

- Започваме с началното състояние $\hat{L}$, за което е ясно, че няма да бъде финално, защото $\varepsilon \notin L$.
- $a^{-1}(L) = \{a,b\}^\star b \stackrel{\text{деф}}{=} M$. Имаме, че $M \neq L$, защото $b \in M$, но $b \notin L$. Тогава

$$\begin{aligned} \delta(\hat{L}, a) &\stackrel{\text{деф}}{=} \hat{M} \quad \text{// защото } a^{-1}(L) = M \\ \delta(\hat{L}, b) &\stackrel{\text{деф}}{=} \hat{\emptyset} \quad \text{// защото } b^{-1}(L) = \emptyset \end{aligned}$$

  $\emptyset$ е съвсем нормален език и напълно нормално да имаме състояние $\hat{\emptyset}$.
- За по-нататък ще е удобно да представим M по следния начин:

$$\begin{aligned} M &= \{a,b\}^\star b \\ &= \{a,b\}^+ b \cup \{b\} \\ &= a \cdot \{a,b\}^\star \cdot b \cup b \cdot \{a,b\}^\star \cdot b \cup \{b\} \\ &= aM \cup bM \cup \{b\}. \end{aligned}$$

  Сега е ясно, че

$$\begin{aligned} a^{-1}(M) &= M \\ b^{-1}(M) &= \{\varepsilon\} \cup M. \end{aligned}$$

  Нека да положим $N \stackrel{\text{деф}}{=} \{\varepsilon\} \cup M$. Имаме, че $N \neq L$ и $N \neq M$, защото $\varepsilon \in N$, но $\varepsilon \notin L$ и $\varepsilon \notin M$. Тогава

$$\begin{aligned} \delta(\hat{M}, a) &\stackrel{\text{деф}}{=} \hat{M} \quad \text{// защото } a^{-1}(M) = M \\ \delta(\hat{M}, b) &\stackrel{\text{деф}}{=} \hat{N} \quad \text{// защото } b^{-1}(M) = N \end{aligned}$$

- Можем да представим езика N по следния начин:

$$N = \{\varepsilon\} \cup aM \cup bM \cup \{b\}.$$

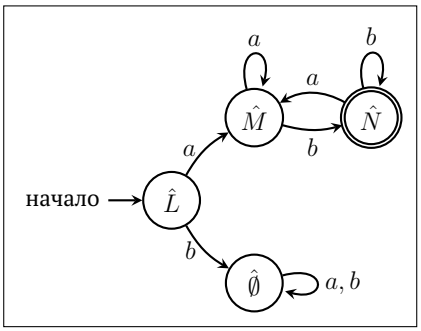
  Тогава имаме, че:

$$\begin{aligned} \delta(\hat{N}, a) &\stackrel{\text{деф}}{=} \hat{M} \quad \text{// защото } a^{-1}(N) = M \\ \delta(\hat{N}, b) &\stackrel{\text{деф}}{=} \hat{N} \quad \text{// защото } b^{-1}(N) = M. \end{aligned}$$

- Завършваме с дефиницията на функцията на преходите като:

$$\begin{aligned} \delta(\hat{\emptyset}, a) &\stackrel{\text{деф}}{=} \hat{\emptyset} \quad \text{// защото } a^{-1}(\emptyset) = \emptyset \\ \delta(\hat{\emptyset}, b) &\stackrel{\text{деф}}{=} \hat{\emptyset} \quad \text{// защото } b^{-1}(\emptyset) = \emptyset. \end{aligned}$$

- Съобразете сами, че $F = \{\hat{N}\}$.



Фигура 2.20: Автомат за езика $\mathcal{L}(\mathbf{a} \cdot (\mathbf{a} + \mathbf{b})^\star \cdot \mathbf{b})$ по метода на Бжозовски.

□

Задача 2.19. Да припомним, че в *Задача 2.4* се искаше да се докаже, че езикът

$$L = \{\alpha \in \{0,1\}^\star \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\}$$

е регулярен. Ние направихме това като построихме автомат за L и доказахме, че той разпознава L. Сега пък постройте автомат за L по метода на Бжозовски.

Решение.

За целта ще ни трябва алтернативна дефиниция на $\overline{\alpha}_{(2)}$. За една дума $\alpha \in \{0,1\}^\star$, можем да дадем следната дефиниция на $\overline{\alpha}_{(2)}$:

- $\overline{\varepsilon}_{(2)} = 0$,
- $\overline{0\alpha}_{(2)} = \overline{\alpha}_{(2)}$,
- $\overline{1\alpha}_{(2)} = 2^{|\alpha|} + \overline{\alpha}_{(2)}$.

Лесно се съобразява, че началното състояние $\hat{L}$ на автомата на Бжозовски не е финално, защото $\varepsilon \notin L$. Да започнем с преходите от началното състояние:

$$\begin{aligned}
0^{-1}(L) &= \{\alpha \in \{0,1\}^\star \mid 0\alpha \in L\} \\
&= \{\alpha \in \{0,1\}^\star \mid \overline{0\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \in \{0,1\}^\star \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= L. \\
1^{-1}(L) &= \{\alpha \in \{0,1\}^\star \mid 1\alpha \in L\} \\
&= \{\alpha \in \{0,1\}^\star \mid \overline{1\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \in \{0,1\}^\star \mid 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&\overset{\text{деф}}{=} M.
\end{aligned}$$

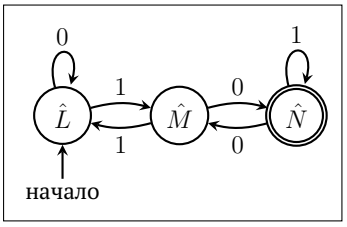
Лесно се съобразява, че езикът M е различен от L, защото например думата $10 \in L$, но $10 \notin M$. Също така, понеже $2^{|\varepsilon|} + \overline{\varepsilon}_{(2)} = 1$, то $\varepsilon \notin M$ и следователно $\hat{M}$ няма да бъде финално състояние в автомата на Бжозовски. Продължаваме нататък:

$$\begin{aligned}
0^{-1}(M) &= \{\alpha \mid 0\alpha \in M\} \\
&= \{\alpha \mid 2^{|0\alpha|} + \overline{0\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&\overset{\text{деф}}{=} N.
\end{aligned}$$

$$\begin{aligned}
1^{-1}(M) &= \{\alpha \mid 1\alpha \in M\} \\
&= \{\alpha \mid 2^{|1\alpha|} + \overline{1\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2 \cdot 2^{|\alpha|} + 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 3 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= L.
\end{aligned}$$

Сега трябва да се уверим, че езикът N е различен от L и M. Това отново е лесно. Непосредствено се проверява, че думата $11 \in N$, но $11 \notin L$ и $11 \notin M$. Също така да отбележим, че понеже $2.2^{|\varepsilon|} + \overline{\varepsilon}_{(2)} = 2$, то $\hat{N}$ ще бъде финално състояние в автомата на Бжозовски, защото $\varepsilon \in N$. Продължаваме нататък с преходите от $\hat{N}$:

$$\begin{aligned}
0^{-1}(N) &= \{\alpha \mid 0\alpha \in N\} \\
&= \{\alpha \mid 2 \cdot 2^{|0\alpha|} + \overline{0\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 4 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 3 \cdot 2^{|\alpha|} + 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= M \\
1^{-1}(N) &= \{\alpha \mid 1\alpha \in N\} \\
&= \{\alpha \mid 2 \cdot 2^{|1\alpha|} + \overline{1\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 4 \cdot 2^{|\alpha|} + 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 3 \cdot 2^{|\alpha|} + 2 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= N.
\end{aligned}$$



Фигура 2.21: Автомат, получен чрез метода на Бжозовски за езика L.

□

Задача 2.20. Постройте автомат по метода на Бжозовски за регулярния език

$$L = \{\omega \in \{a,b\}^\star \mid |\omega|_a \text{ е четно и } |\omega|_b = 1\}.$$

Решение.

Да започнем с преходите от началното състояние $\hat{L}$:

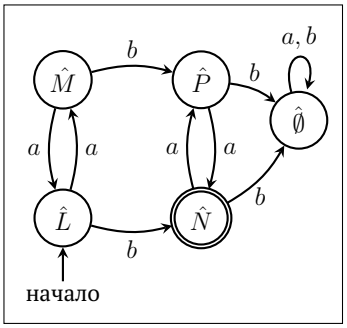
$$a^{-1}(L) = \{\omega \in \{a,b\}^\star \mid |\omega|_a \text{ е нечет. и } |\omega|_b = 1\} \stackrel{\text{деф}}{=} M$$
$$b^{-1}(L) = \{\omega \in \{a,b\}^\star \mid |\omega|_a \text{ е четно и } |\omega|_b = 0\} \stackrel{\text{деф}}{=} N.$$

Лесно се вижда, че M и N са различни езици от L. Например, $aab \in L$, но $aab \notin M$; $aa \in N$, но $aa \notin L$ и $aa \notin M$. Сега да видим какви преходи имаме от състоянието $\hat{M}$:

$$a^{-1}(M) = L;$$
$$b^{-1}(M) = \{\omega \in \{a,b\}^\star \mid |\omega|_a \text{ е нечет. и } |\omega|_b = 0\} \stackrel{\text{деф}}{=} P.$$

Ясно е, че $P \neq M, L, N$, защото $a \in P$, но $a \notin M, L, N$. Завършваме с преходите от състоянията $\hat{N}$ и $\hat{P}$:

$$a^{-1}(N) = P \quad \text{и} \quad b^{-1}(N) = \emptyset$$
$$a^{-1}(P) = N \quad \text{и} \quad b^{-1}(P) = \emptyset.$$



Фигура 2.22: Автомат, който приема думи с четен брой a и точно едно b, получен чрез метода на Бжозовски.

□

Ние формално не сме дефинирали понятието *безкраен* детерминиран автомат. Този пример се разглежда и в [Sak09, стр. 113].

Пример 2.8. Да разгледаме езика $L = \{a^n b^n \mid n \in \mathbb{N}\}$. Ние вече знаем от *Пример 2.6*, че L не е регулярен език. Да се опитаме да построим автомат, който го разпознава.

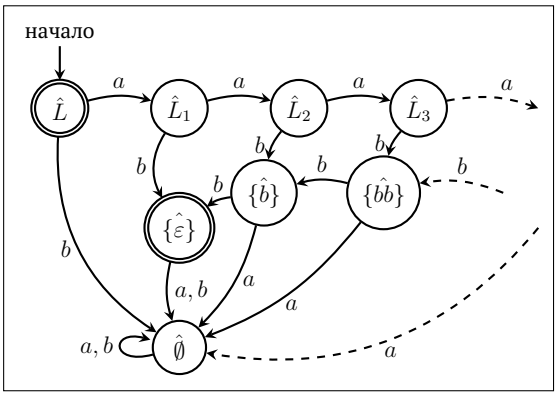
Нека за всяко k да разгледаме езика

$$L_k \stackrel{\text{деф}}{=} \{a^n b^{n+k} \mid n \in \mathbb{N}\}.$$

Да видим какво се получава като приложим процедурата за строене на минимален автомат.

- $a^{-1}(L) = L_1$;
- $b^{-1}(L) = \emptyset$;
- $a^{-1}(L_1) = L_2$;
- $b^{-1}(L_1) = \{\varepsilon\}$;
- $a^{-1}(\{\varepsilon\}) = b^{-1}(\{\varepsilon\}) = \emptyset$;
- Лесно можем да докажем, че за всяко k е изпълнено, че $a^{-1}(L_k) = L_{k+1}$.
- Лесно се вижда, че $b^{-1}(L_{k+1}) = \{b^k\}$, за всяко k.
- Ясно е, че $b^{-1}(\{b^k\}) = \{b^{k-1}\}$, за $k \geq 1$.

Получаваме, че езикът L се разпознава от автомат с *безкрайно много състояния*.



Фигура 2.23: *Безкраен* автомат за езика $\{a^n b^n \mid n \in \mathbb{N}\}$ построен по метода на Бжозовски.

2.8 Минимален автомат

Сега ще видим, че автоматът на Бжозовски $\mathcal{B}$ за даден регулярен език L е в известен смисъл най-добрият възможен. Накратко, $\mathcal{B}$ има най-малкия възможен брой състояния измежду всички детерминирани крайни автомати, които разпознават L. За да успеем да видим това, първо трябва да се подготвим.

Нека $\mathcal{A} = \langle \Sigma, Q, q_{\mathtt{start}}, \delta, F \rangle$ е ДКА. За всяко състояние q на $\mathcal{A}$ да разгледаме езика

$$\mathcal{L}_{\mathcal{A}}(q) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \delta^\star(q, \alpha) \in F\}.$$

В частност имаме, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(q_{\mathtt{start}})$. Без ограничение на общността, нека приемем, че винаги разглеждаме само свързани ДКА $\mathcal{A}$, т.е. всяко състояние е достижимо от началното. Нека за всяка дума α да положим $q_\alpha \stackrel{\text{деф}}{=} \delta^\star(q_{\mathtt{start}}, \alpha)$. Понеже $\mathcal{A}$ е свързан, то всяко състояние на $\mathcal{A}$ може да се разглежда като q_α за някоя дума α.

Тук използваме, че δ е тотална функция. За някои състояния p може да съществуват безкрайно много думи α, за които $q_\alpha = p$.

Твърдение 2.15. Нека $L = \mathcal{L}(\mathcal{A})$. Тогава за всяка дума α е изпълнено, че:

$$\mathcal{L}_{\mathcal{A}}(q_\alpha) = \alpha^{-1}(L).$$

Доказателство. За произволна дума γ имаме следните еквивалентности:

$$\begin{aligned}
\gamma \in \mathcal{L}_{\mathcal{A}}(q_\alpha) &\Leftrightarrow \delta^\star(q_\alpha, \gamma) \in F && \text{// от деф. на } \mathcal{L}_{\mathcal{A}}(q_\alpha) \\
&\Leftrightarrow \delta^\star(\delta^\star(q_{\mathtt{start}}, \alpha), \gamma) \in F && \text{// } q_\alpha \stackrel{\text{деф}}{=} \delta^\star(q_{\mathtt{start}}, \alpha) \\
&\Leftrightarrow \delta^\star(q_{\mathtt{start}}, \alpha\gamma) \in F && \text{// } \textit{Твърдение 2.1} \\
&\Leftrightarrow \alpha\gamma \in \mathcal{L}(\mathcal{A}) && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\
&\Leftrightarrow \gamma \in \alpha^{-1}(L). && \text{// } L = \mathcal{L}(\mathcal{A})
\end{aligned}$$

□

Твърдение 2.16. Нека $\mathcal{B}$ е автомат на Бжозовски за езика L. Тогава за произволно състояние $\hat{M}$ на $\mathcal{B}$,

$$\mathcal{L}_{\mathcal{B}}(\hat{M}) = M.$$

Доказателство. Понеже за произволна дума α имаме еквивалентностите

$$\begin{aligned}
\alpha \in \mathcal{L}_{\mathcal{B}}(\hat{M}) &\Leftrightarrow \delta^\star_{\mathcal{B}}(\hat{M}, \alpha) \in F^{\mathcal{B}} \\
&\Leftrightarrow \varepsilon \in \alpha^{-1}(M) \\
&\Leftrightarrow \alpha \in M,
\end{aligned}$$

заключаваме, че $\mathcal{L}_{\mathcal{B}}(\hat{M}) = M$. □

Твърдение 2.17. Нека A и B са множества, като A е крайно, за които съществува *сюрективна* функция $f : A \to B$. Тогава $|B| \leq |A|$.

Вярно ли е това твърдение, ако A е безкрайно ?

Упътване. Понеже A е крайно множество, можем да изброим елементите му в редица. Нека $A = \{a_0, a_1, \ldots, a_{n-1}\}$. Разгледайте $g : B \to A$, където

$$g(b) \stackrel{\text{деф}}{=} a_m \text{ за } m = \min\{i < n \mid f(a_i) = b\}.$$

Да отбележим, че дефиницията на g е коректна, защото множеството $\{i < n \mid f(a_i) = b\}$ е непразно, понеже f е сюрективна. Докажете, че g е инективна. □

Лема 2.7. Нека L е регулярен език и $\mathcal{A}$ е ДКА, който разпознава L, а $\mathcal{B}$ е автоматът на Бжозовски за L. Тогава $|Q^{\mathcal{B}}| \leq |Q^{\mathcal{A}}|$.

Тази лема ни казва, че автоматът на Бжозовски има възможно най-малкия брой състояния измежду всички ДКА разпознаващи L.

Доказателство. Да разгледаме функцията $f : Q^{\mathcal{A}} \to Q^{\mathcal{B}}$ зададена по следния начин:

$$f(q) \stackrel{\text{деф}}{=} \hat{M}, \text{ където } M = \mathcal{L}_{\mathcal{A}}(q).$$

Нека първо да съобразим защо f е добре дефинирана функция. Да напомним, че приехме, че $\mathcal{A}$ е свързан автомат. Тогава за произволно състояние p, нека α е една дума, за която $p = \delta^\star(q_{\texttt{start}}, \alpha)$, т.е. $p = q_\alpha$ според означението, което въведохме в началото на *Раздел 2.8*. Тогава от *Твърдение 2.15* следва, че $f(q_\alpha) = \hat{M}$, където $M = \alpha^{-1}(L)$, защото $\alpha^{-1}(L) = \mathcal{L}_{\mathcal{A}}(q_\alpha)$.

Освен това, f е сюрективна, защото, според дефиницията на автомат на Бжозовски, за произволно състояние $\hat{M} \in Q^{\mathcal{B}}$ има дума α, за която $M = \alpha^{-1}(L)$. Тогава $f(q_\alpha) = \hat{M}$. Сега от *Твърдение 2.17* можем да заключим, че

$$|Q^{\mathcal{B}}| \leq |Q^{\mathcal{A}}|.$$

□

Така получаваме следния критерий за проверка дали даден език е регулярен.

Следствие 2.6. Един език L е регулярен точно тогава, когато автоматът $\mathcal{B}$ на Бжозовски за L, има крайно много състояния.

Доказателство. Нека L е регулярен. Тогава $L = \mathcal{L}(\mathcal{A})$ за някой ДКА $\mathcal{A}$. Да напомним, че от *Твърдение 2.14* също имаме и $\mathcal{L}(\mathcal{B}) = L$. Понеже $|Q^{\mathcal{B}}| \leq |Q^{\mathcal{A}}|$, то $\mathcal{B}$ има краен брой състояния. Обратно, ако $\mathcal{B}$ има крайно много състояния, то тогава $\mathcal{B}$ е ДКА. Понеже $\mathcal{L}(\mathcal{B}) = L$, то L е автоматен и следователно регулярен. □

Следствие 2.7. Нека L е регулярен език. Тогава автоматът $\mathcal{B}$, построен по метода на Бжозовски за L, има минималния възможен брой състояния измежду всички детерминирани крайни автомати разпознаващи L.

Твърдение 2.18. Нека A и B са крайни равномощни множества. Докажете, че ако $g : A \to B$ е сюрекция, то g е биекция.

Доказателство. Нека $A = \{a_0, \ldots, a_{n-1}\}$ и $B = \{b_0, \ldots, b_{n-1}\}$. Нека, за всеки индекс $i < n$ да положим

Ясно е, че щом A и B са равномощни, то има биекция между тях. Тук доказваме, че всяка сюрекция между тях е също така и биекция.

$$A_i \stackrel{\text{деф}}{=} \{a_j \in A \mid g(a_j) = b_i\}.$$

Щом g е сюрекция, то $A_i \neq \emptyset$ за всеки индекс $i < n$. Понеже g е функция, то $A_i \cap A_j = \emptyset$ за всеки два различни индекса i и j. Това означава, че

Да напомним, че от курса по Дискретна математика имаме формулата $|X \cup Y| = |X| + |Y| + |X \cap Y|$. Просто в нашия случай $|X \cap Y| = 0$.

$$n = |A| = |\bigcup_{i<n} A_i| = \sum_{i<n} |A_i|.$$

Оттук следва, че щом за всяко i имаме, че $|A_i| \neq 0$, то $|A_i| = 1$. Заключаваме, че g е инекция, защото в противен случай щяхме да имаме някое i, за което $|A_i| > 1$. □

Теорема 2.3. За всеки регулярен език L съществува единствен минимален ДКА с точност до изоморфизъм.

Доказателство. Вече знаем, че автоматът $\mathcal{B}$, построен по метода на Бжозовски за L, има минималния възможен брой състояния. Нека $\mathcal{A}$ е друг ДКА разпознаващ L и $|Q^{\mathcal{A}}| = |Q^{\mathcal{B}}|$. Трябва да докажем, че $\mathcal{A} \cong \mathcal{B}$. Ясно е, че $\mathcal{A}$ е свързан автомат, т.е. всяко състояние p на $\mathcal{A}$ е от вида $p = q_\alpha$. В противен случай, $\mathcal{A}$ нямаше да бъде минимален. Да разгледаме функцията $f : Q^{\mathcal{A}} \to Q^{\mathcal{B}}$, където

С други думи, ако имаме два минимални ДКА за L, то можем да получим единия автомат от другия чрез внимателно преименуване на състоянията.

$$f(p) = \hat{M} \stackrel{\text{деф}}{\Leftrightarrow} M = \mathcal{L}_{\mathcal{A}}(p).$$

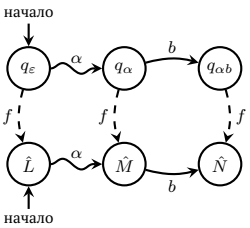
От *Твърдение 2.15* знаем, че $f(q_\alpha) = \mathcal{L}_{\mathcal{A}}(q_\alpha) = \alpha^{-1}(L)$ следва, че f е сюрективна. Понеже $|Q^{\mathcal{A}}| = |Q^{\mathcal{B}}|$, то от *Твърдение 2.18* имаме, че f е всъщност биекция. Остава да видим защо $\mathcal{A} \cong_f \mathcal{B}$. Да напомним, че състоянията на $Q^{\mathcal{A}}$ можем да ги разглеждаме като q_α и тогава $\mathcal{L}_{\mathcal{A}}(q_\alpha) = \alpha^{-1}(L)$ от *Твърдение 2.15*.

- За началното състояние имаме, че:

$$\begin{aligned} f(q^{\mathcal{A}}_{\mathtt{start}}) &= f(q_\varepsilon) && \text{// } q^{\mathcal{A}}_{\mathtt{start}} = q_\varepsilon \\ &= \hat{L}. && \text{// } \varepsilon^{-1}(L) = L \end{aligned}$$

- За финалните състояния имаме, че:

$$\begin{aligned} q_\alpha \in F^{\mathcal{A}} &\Leftrightarrow \delta^\star_{\mathcal{A}}(q_{\mathtt{start}}, \alpha) \in F^{\mathcal{A}} && \text{// } q_\alpha = \delta^\star_{\mathcal{A}}(q_{\mathtt{start}}, \alpha) \\ &\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}) && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \varepsilon \in \alpha^{-1}(L) && \text{// } L = \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \hat{M} \in F^{\mathcal{B}}. && \text{// за } M = \alpha^{-1}(L) \end{aligned}$$



- Остава да докажем, че за произволна буква b и произволни състояния $q, p \in Q_1$ е изпълнено, че:
$$\delta_{\mathcal{A}}(q, b) = p \;\Leftrightarrow\; \delta_{\mathcal{B}}(f(q), b) = f(p). \tag{2.9}$$
Понеже всяко състояние на $\mathcal{A}$ може да се запише като q_α за някоя дума α, то можем да запишем горния ред и така:
$$\delta_{\mathcal{A}}(q_\alpha, b) = p \;\Leftrightarrow\; \delta_{\mathcal{B}}(f(q_\alpha), b) = f(p).$$
Нека положим
$$M \stackrel{\text{деф}}{=} \alpha^{-1}(L)$$
$$N \stackrel{\text{деф}}{=} (\alpha b)^{-1}(L) = b^{-1}(\alpha^{-1}(L)) = b^{-1}(M).$$
От *Твърдение 2.15* знаем, че $\hat{M} = f(q_\alpha)$, защото $\mathcal{L}_{\mathcal{A}}(q_\alpha) = \alpha^{-1}(L)$. За посоката $(\Rightarrow)$, нека разгледаме състоянието p, за което $\delta_{\mathcal{A}}(q_\alpha, b) = p$. Ясно е, че $p = q_{\alpha b}$, защото
$$p = \delta_{\mathcal{A}}(q_\alpha, b) = \delta_{\mathcal{A}}(\delta^\star_{\mathcal{A}}(q^{\mathcal{A}}_{\mathtt{start}}, \alpha), b) = \delta^\star_{\mathcal{A}}(q^{\mathcal{A}}_{\mathtt{start}}, \alpha b).$$
Тогава имаме, че $f(p) = f(q_{\alpha b}) = \hat{N}$, защото $(\alpha b)^{-1}(L) = \mathcal{L}_{\mathcal{A}}(q_{\alpha b})$ отново според *Твърдение 2.15*. Остава единствено да отбележим, че от конструкцията на автомата $\mathcal{B}$ знаем, че:
$$\delta_{\mathcal{B}}(\underbrace{\hat{M}}_{f(q_\alpha)}, b) = \underbrace{\hat{N}}_{f(p)}.$$
Така получихме дясната страна на еквивалентността в Свойство 2.9.

За посоката $(\Leftarrow)$, нека p е състояние, за което $\delta_{\mathcal{B}}(f(q_\alpha), b) = f(p)$. Според конструкцията на автомата $\mathcal{B}$, $\delta_{\mathcal{B}}(\hat{M}, b) = \hat{N} = f(p)$. Знаем също, че $N = \mathcal{L}_{\mathcal{A}}(p) = (\alpha b)^{-1}(L) = \mathcal{L}_{\mathcal{A}}(q_{\alpha b})$. Понеже f е биекция, то $p = q_{\alpha b}$, защото $\mathcal{L}_{\mathcal{A}}(p) = \mathcal{L}_{\mathcal{A}}(q_{\alpha b})$. Заключаваме, че
$$\delta_{\mathcal{A}}(q_\alpha, b) = p.$$

□

Примерни задачи

За да докажем, че един език L не е регулярен можем да приложим *Следствие 2.6* като докажем, че автоматът на Бжозовски за L има безкрайно много състояния. Обърнете внимание, че не е нужно да намерим всички състояния на автомата $\mathcal{B}$, а само това, че са безкрайно много.

Пример 2.9. За езика $L = \{a^n b^n \mid n \in \mathbb{N}\}$ имаме, че за произволни естествени числа k и m е изпълнено следното свойство: Сравнете с *Пример 2.8*.

$$k < m \implies (a^k)^{-1}(L) \neq (a^m)^{-1}(L).$$

Проверете, че $(a^k)^{-1}(L) = \{a^n b^{n+k} \mid n \in \mathbb{N}\}$, за всяко $k \in \mathbb{N}$. Така получаваме, че автоматът на Бжозовски за L ще има безкрайно много състояния. Заключаваме, че този език **не** е регулярен.

Пример 2.10. За езика $L = \{a^{n^2} \mid n \in \mathbb{N}\}$ имаме, че за произволни естествени числа k и m е изпълнено следното свойство:

$$k < m \implies (a^{k^2})^{-1}(L) \neq (a^{m^2})^{-1}(L).$$

За да покажем, че $(a^{k^2})^{-1}(L) \neq (a^{m^2})^{-1}(L)$ е достатъчно да посочим дума γ, за която $\gamma \in (a^{k^2})^{-1}(L)$, но $\gamma \notin (a^{m^2})^{-1}(L)$. Да разгледаме думата $\gamma = a^{2k+1}$. Ясно е, че $\gamma \in (a^{k^2})^{-1}(L)$, защото $a^{k^2}\gamma = a^{(k+1)^2} \in L$, но понеже $k < m$, то

$$m^2 < m^2 + 2k + 1 < m^2 + 2m + 1 = (m+1)^2$$

и следователно $\gamma \notin (a^{m^2})(L)$, защото $a^{m^2}\gamma = a^{m^2+2k+1} \notin L$. Заключаваме, че този език **не** е регулярен.

Пример 2.11. За езика $L = \{a^{n!} \mid n \in \mathbb{N}\}$ имаме, че за произволни естествени числа k и m е изпълнено следното свойство:

$$k < m \implies (a^{k!})^{-1}(L) \neq (a^{m!})^{-1}(L).$$

Да разгледаме $k < m$ и думата $\gamma = a^{(k!)k}$. Тогава $\gamma \in (a^{k!})^{-1}(L)$, защото $a^{k!}\gamma = a^{k!+(k!)k} = a^{(k+1)!} \in L$, но

$$m! < m! + (k!)k < m! + (m!)m = (m+1)!$$

и следователно $\gamma \notin (a^{m!})^{-1}(L)$, защото $a^{m!}\gamma = a^{m!+(k!)k} \notin L$. Заключаваме, че този език **не** е регулярен.

Задача 2.21. Докажете, че езикът

$$L = \{a^{f_n} \mid f_0 = f_1 = 1 \,\&\, f_{n+2} = f_{n+1} + f_n\}$$

не е регулярен.

2.9 Автомат на Майхил-Нероуд

на англ. Myhill-Nerode

Нека L е език и нека α и β са думи. Казваме, че α и β са **еквивалентни относно** L, което записваме като $\alpha \approx_L \beta$, когато:

$$\alpha \approx_L \beta \overset{\text{деф}}{\Leftrightarrow} \alpha^{-1}(L) = \beta^{-1}(L).$$

С други думи,

$$\alpha \approx_L \beta \Leftrightarrow (\forall \omega \in \Sigma^\star)[\, \alpha\omega \in L \Leftrightarrow \beta\omega \in L \,].$$

$\approx_L$ е известна като релация на Майхил-Нероуд

Задача 2.22. Докажете, че за всяка дума α е изпълнено, че:

$$\alpha \in L \Leftrightarrow [\alpha]_L \subseteq L.$$

Ще наричаме $\mathcal{M}$ автомат на Майхил-Нероуд. На практика във всеки учебник се разглежда автомата на Майхил-Нероуд вместо автомата на Бжозовски. Например, [PL98, стр. 98], [HU79, стр. 65], [Sip12, стр. 91], [Koz97, стр. 89]

Ще дефинираме детерминиран автомат $\mathcal{M} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ по следния начин:

- $Q \overset{\text{деф}}{=} \{\, [\alpha]_L \mid \alpha \in \Sigma^\star \,\}$;
- $q_{\texttt{start}} \overset{\text{деф}}{=} [\varepsilon]_L$;
- $F \overset{\text{деф}}{=} \{\, [\alpha]_L \mid \alpha \in L \,\}$;
- Определяме функцията на преходите δ като за всяка буква b и всяка дума α,

$$\delta([\alpha]_L, b) \overset{\text{деф}}{=} [\alpha b]_L.$$

С други думи, трябва да проверим, че нашата дефиниция на δ не зависи от избора на представител от класа $[\alpha]_L$, който сме направили.

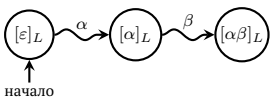
Задача 2.23. Докажете, че $\delta : Q^{\mathcal{M}} \times \Sigma \to Q^{\mathcal{M}}$ е добре дефинирана функция.

Упътване. Трябва да докажете, че

$$[\alpha]_L = [\beta]_L \implies \delta([\alpha]_L, b) = \delta([\beta]_L, b).$$

□

Задача 2.24. Докажете, че ако $\mathcal{M}$ е автоматът на Майхил-Нероуд за езика L, то $\mathcal{L}(\mathcal{M}) = L$.



Упътване. Докажете с индукция по дължината на думата β, че е изпълнено свойството $(\forall \beta \in \Sigma^\star) P(\beta)$, където

$$P(\beta) \overset{\text{деф}}{=} (\forall \alpha \in \Sigma^\star)[\delta^\star_{\mathcal{M}}([\alpha]_L, \beta) = [\alpha\beta]_L].$$

Оттук заключете директно, че $\mathcal{L}(\mathcal{M}) = L$. □

Задача 2.25. Да разгледаме един език L. Нека $\mathcal{B}$ е автоматът на Бжозовски за L и $\mathcal{M}$ е автоматът на Майхил-Нероуд за L. Да разгледаме $f : Q^{\mathcal{M}} \to Q^{\mathcal{B}}$, където:

Обърнете внимание, че тук не изискваме L да е регулярен.

$$f([\alpha]_L) = \hat{K} \overset{\text{деф}}{\Leftrightarrow} K = \alpha^{-1}(L).$$

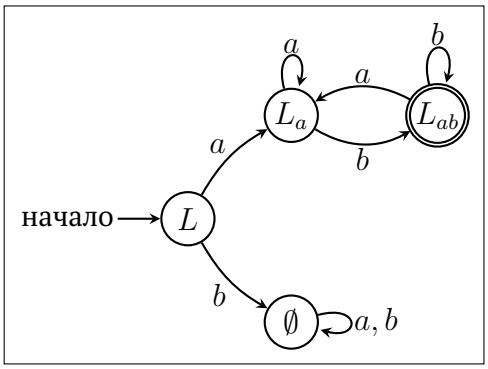
Докажете, че f е биекция.

Теоремата е доказана независимо от Майхил [Myh57] и Нероуд [Ner58].

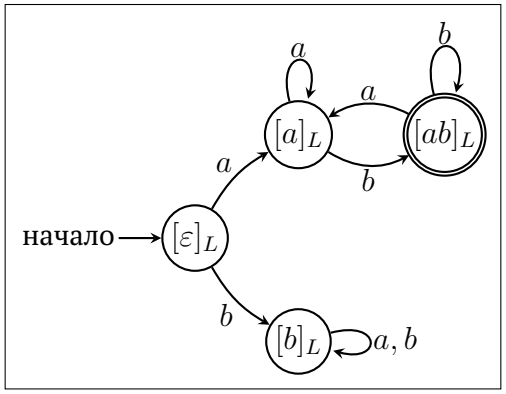
Задача 2.26 (Теорема на Майхил-Нероуд)**.** Докажете, че L е регулярен език точно тогава, когато автоматът на Майхил-Нероуд $\mathcal{M}$ е краен. Освен това, докажете, че $\mathcal{M}$ е минимален ДКА за L.

Пример 2.12. Да разгледаме езика $L = \mathcal{L}(\mathbf{a} \cdot (\mathbf{a} + \mathbf{b})^\star \cdot \mathbf{b})$.

Да напомним, че в *Пример 2.18* вече видяхме автоматът на Бжозовски за L. Тук пишем L_a и L_{ab} вместо $a^{-1}(L)$ и $(ab)^{-1}(L)$ за да спестим място.

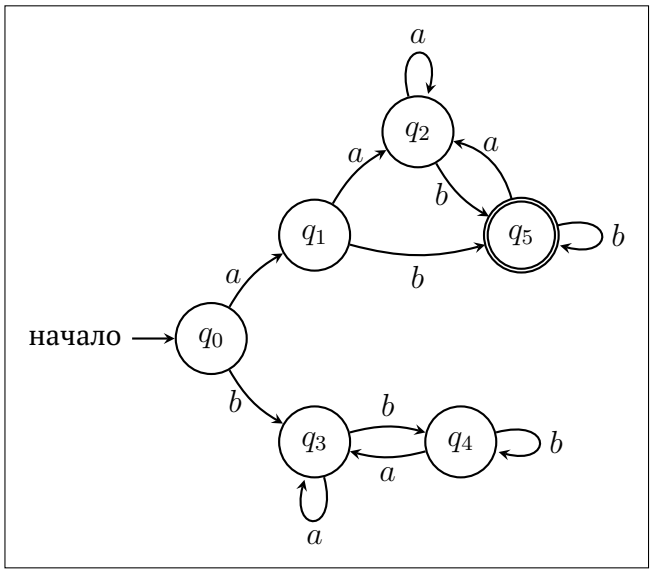


(а) Автомат за L по метода на Бжозовски.



(б) Автомат за L по метода на Майхил-Нероуд.

Пример 2.13. Нека сега да приемем, че не знаем минималния автомат за езика L от предишния пример, а имаме следния автомат $\mathcal{A}$ за L:



Веднага се вижда, че $\mathcal{L}_{\mathcal{A}}(q_3) = \mathcal{L}_{\mathcal{A}}(q_4) = \emptyset$. Друг начин да се види равенството е по следния начин:

$$\mathcal{L}_{\mathcal{A}}(q_3) = \{a\} \cdot \mathcal{L}_{\mathcal{A}}(q_3) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(q_4);$$
$$\mathcal{L}_{\mathcal{A}}(q_4) = \{a\} \cdot \mathcal{L}_{\mathcal{A}}(q_3) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(q_4).$$

Това означава, че в минималния автомат състоянията q_3 и q_4 ще бъдат заменени от едно състояние.

Лесно се вижда също, че $\mathcal{L}_{\mathcal{A}}(q_1) = \mathcal{L}_{\mathcal{A}}(q_2)$. Това е така, защото

$$\mathcal{L}_{\mathcal{A}}(q_1) = \{a\} \cdot \mathcal{L}_{\mathcal{A}}(q_2) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(q_5);$$
$$\mathcal{L}_{\mathcal{A}}(q_2) = \{a\} \cdot \mathcal{L}_{\mathcal{A}}(q_2) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(q_5).$$

Естествено, при по-сложни автомати, ние няма да можем толкова лесно да съобразим кои състояния може да „слеем”. Затова сега ще изучим този въпрос малко по-задълбочено.

2.10 Минимизация

- Нека отново да приемем, че сме фиксирали един детерминиран краен автомат $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ като всяко състояние е достижимо от началното. Да напомним, че в началото на Раздел 2.8 с $\mathcal{L}_{\mathcal{A}}(p)$ означаваме езикът, който се разпознава от автомата $\mathcal{A}$, ако приемем, че p е началното състояние, т.е. $\mathcal{L}_{\mathcal{A}}(p) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \delta^\star(p, \omega) \in F\}$. В частност, $\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}})$.

- Сега дефинираме следната релация между състояния на автомата $\mathcal{A}$:

$$p \equiv_{\mathcal{A}} q \stackrel{\text{деф}}{\Leftrightarrow} \mathcal{L}_{\mathcal{A}}(p) = \mathcal{L}_{\mathcal{A}}(q).$$

  Това означава, че $p \equiv_{\mathcal{A}} q$ точно тогава, когато

$$(\forall \omega \in \Sigma^\star)[\, \delta^\star(p, \omega) \in F \;\Leftrightarrow\; \delta^\star(q, \omega) \in F \,]. \tag{2.10}$$

- Релацията $\equiv_{\mathcal{A}}$ между състояния на автомата $\mathcal{A}$ е релация на еквивалентност. ✍ Защо?

Твърдение 2.19. За произволно състояние q е изпълнено, че:

$$[q]_{\equiv_{\mathcal{A}}} \cap F \neq \emptyset \;\Leftrightarrow\; [q]_{\equiv_{\mathcal{A}}} \subseteq F.$$

Доказателство. Посоката ($\Leftarrow$) е очевидна. Щом $p \in F$, то $\varepsilon \in \mathcal{L}_{\mathcal{A}}(p)$. Щом $p \equiv_{\mathcal{A}} q$, то $\mathcal{L}_{\mathcal{A}}(p) = \mathcal{L}_{\mathcal{A}}(q)$ и следователно $\varepsilon \in \mathcal{L}_{\mathcal{A}}(q)$. С други думи, $\delta^\star_{\mathcal{A}}(q, \varepsilon) \in F$, откъдето получаваме, че $q \in F$. □

Сравнете с *Задача 2.22*.

Твърдение 2.20. За произволни състояния p и q и произволна буква a е изпълнено:

$$p \equiv_{\mathcal{A}} q \implies \delta(p, a) \equiv_{\mathcal{A}} \delta(q, a).$$

Доказателство. Да разгледаме произволни състояния p, q и буква a. Тогава:

$$\begin{aligned}
p \equiv_{\mathcal{A}} q &\Leftrightarrow \mathcal{L}_{\mathcal{A}}(p) = \mathcal{L}_{\mathcal{A}}(q) \\
&\Leftrightarrow (\forall \beta \in \Sigma^\star)[\delta^\star(p, \beta) \in F \;\Leftrightarrow\; \delta^\star(q, \beta) \in F] \\
&\Rightarrow (\forall \beta \in \Sigma^\star)[\delta^\star(p, a\beta) \in F \;\Leftrightarrow\; \delta^\star(q, a\beta) \in F] \\
&\Leftrightarrow (\forall \beta \in \Sigma^\star)[\delta^\star(\delta(p, a), \beta) \in F \;\Leftrightarrow\; \delta^\star(\delta(q, a), \beta) \in F] \\
&\Leftrightarrow \mathcal{L}_{\mathcal{A}}(\delta(p, a)) = \mathcal{L}_{\mathcal{A}}(\delta(q, a)) \\
&\Leftrightarrow \delta(p, a) \equiv_{\mathcal{A}} \delta(q, a).
\end{aligned}$$

□

За автомата $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$, дефинираме автомата

$$\mathcal{A}' = \langle \Sigma, Q', q'_{\texttt{start}}, \delta', F' \rangle$$

по следния начин:

- $Q' \stackrel{\text{деф}}{=} \{[q]_{\equiv_\mathcal{A}} \mid q \in Q\}$;
- $q'_{\texttt{start}} \stackrel{\text{деф}}{=} [q_{\texttt{start}}]_{\equiv_\mathcal{A}}$;
- $\delta'([q]_{\equiv_\mathcal{A}}, a) \stackrel{\text{деф}}{=} [\delta(q,a)]_{\equiv_\mathcal{A}}$;
- $F' \stackrel{\text{деф}}{=} \{[q]_{\equiv_\mathcal{A}} \mid [q]_{\equiv_\mathcal{A}} \cap F \neq \emptyset\}$;

Задача 2.27. Докажете, че $\delta' : Q' \times \Sigma \to \Sigma$ е добре дефинирана функция.

С други думи, дефиницията на δ' не зависи от избора на представител от класа $[q]_{\equiv_\mathcal{A}}$, който сме направили. Сравнете с *Задача 2.23*.

Упътване. От *Твърдение 2.20* имаме, че

$$[p]_{\equiv_\mathcal{A}} = [q]_{\equiv_\mathcal{A}} \implies \delta'([p]_{\equiv_\mathcal{A}}, a) = \delta'([q]_{\equiv_\mathcal{A}}, a).$$

□

Твърдение 2.21. За всяко състояние q на автомата $\mathcal{A}$ и дума α е изпълнено, че

$$\delta'^\star([q]_{\equiv_\mathcal{A}}, \alpha) = [\delta^\star(q,\alpha)]_{\equiv_\mathcal{A}}. \tag{2.11}$$

Доказателство. Ще докажем Свойство (2.11) с индукция по дължината на α.

- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Тогава

$$\delta'^\star([q]_{\equiv_\mathcal{A}}, \varepsilon) = [q]_{\equiv_\mathcal{A}} = [\delta^\star(q,\varepsilon)]_{\equiv_\mathcal{A}}.$$

- Да приемем, че Свойство (2.11) е вярно за думи с дължина n.
- Нека $|\alpha| = n+1$, т.е. $\alpha = \beta a$ и $|\beta| = n$. Тогава

$$\begin{aligned}
\delta'^\star([q]_{\equiv_\mathcal{A}}, \beta a) &= \delta'(\delta'^\star([q]_{\equiv_\mathcal{A}}, \beta), a) && \text{// деф. на } \delta'^\star \\
&= \delta'(\underbrace{[\delta^\star(q,\beta)]_{\equiv_\mathcal{A}}}_{p}, a) && \text{// И.П. за } \beta \\
&= \delta'([p]_{\equiv_\mathcal{A}}, a) \\
&= [\delta(p,a)]_{\equiv_\mathcal{A}} && \text{// деф. на } \delta' \\
&= [\delta(\delta^\star(q,\beta), a)]_{\equiv_\mathcal{A}} && \text{// } p = \delta'^\star(q,\beta) \\
&= [\delta^\star(q,\beta a)]_{\equiv_\mathcal{A}}. && \text{// деф. на } \delta^\star
\end{aligned}$$

□

Лема 2.8. $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$.

Доказателство. Достатъчно е да проследим следните еквивалентности за произволна дума α:

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}) &\Leftrightarrow \delta^\star(q_{\texttt{start}}, \alpha) \in F && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\
&\Leftrightarrow [\delta^\star(q_{\texttt{start}}, \alpha)]_{\equiv_\mathcal{A}} \in F' && \text{// деф. на } F' \\
&\Leftrightarrow \delta'^\star([q_{\texttt{start}}]_{\equiv_\mathcal{A}}, \alpha) \in F' && \text{// } \textit{Твърдение 2.21} \\
&\Leftrightarrow \delta'^\star(q'_{\texttt{start}}, \alpha) \in F' && \text{// } q'_{\texttt{start}} \stackrel{\text{деф}}{=} [q_{\texttt{start}}]_{\equiv_\mathcal{A}} \\
&\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}'). && \text{// деф. на } \mathcal{L}(\mathcal{A}')
\end{aligned}$$

□

Остава да докажем, че автоматът $\mathcal{A}'$ е минимален за езика $\mathcal{L}(\mathcal{A})$. За да направим това ни трябва едно помощно твърдение.

Твърдение 2.22. Нека A и B са множества и $f : A \to B$ е сюрекция. Дефинираме релация на еквивалентност $\equiv$ между елементи на A по следния начин:

$$a_0 \equiv a_1 \stackrel{\text{деф}}{\Leftrightarrow} f(a_0) = f(a_1).$$

Дефинираме $[a]_\equiv$ да бъде класът на еквивалентност на a, т.е.

$$[a]_\equiv \stackrel{\text{деф}}{=} \{a_0 \in A \mid f(a) = f(a_0)\}.$$

Нека $A' \stackrel{\text{деф}}{=} \{[a]_\equiv \mid a \in A\}$. Тогава $f' : A' \to B$, където $f'([a]_\equiv) \stackrel{\text{деф}}{=} f(a)$ е биекция.

Това твърдение го формулираме в общия случай, но ние ще го използваме, когато $\equiv$ е релацията $\equiv_\mathcal{A}$. Сравнете с *Твърдение 2.18*.

Доказателство. Достатъчно е да направим следните проверки:

- Ако $[a_0]_\equiv = [a_1]_\equiv$, то $f(a_0) = f(a_1)$ и следователно $f'([a_0]_\equiv) = f'([a_1]_\equiv)$. Това означава, че f' е функция.
- Понеже f е сюрекция, за произволен елемент $b \in B$, съществува $a \in A$, за който $f(a) = b$. По дефиниция, $f'([a]_\equiv) = b$. Това означава, че f' е сюрекция.
- Нека сега $[a_0]_\equiv \neq [a_1]_\equiv$, т.е. $f(a_0) \neq f(a_1)$. Тогава $f'([a_0]_\equiv) \neq f'([a_1]_\equiv)$. Това означава, че f' е инекция.

От всичко това заключаваме, че f' е биекция. □

Теорема 2.4. Автоматът $\mathcal{A}'$ е минимален ДКА за езика $\mathcal{L}(\mathcal{A})$.

Доказателство. Нека $\mathcal{B}$ е автоматът на Бжозовски за езика $\mathcal{L}(\mathcal{A})$. За да докажем, че $\mathcal{A}'$ е минимален ДКА за $\mathcal{L}(\mathcal{A})$, според *Лема 2.8* и *Теорема 2.3*, достатъчно е да докажем,

Възможно е в доказателството да подходим и по друг начин. Ако докажем, че $\mathcal{A}' \cong_{f'} \mathcal{B}$, то оттук директно следва, че $\mathcal{A}'$ е минимален автомат разпознаващ L.

че $|Q'| = |Q^{\mathcal{B}}|$. От доказателството на *Лема 2.7* знаем, че функцията $f : Q^{\mathcal{A}} \to Q^{\mathcal{B}}$ е сюрекция, където

$$f(q) = \hat{M} \overset{\text{деф}}{\Leftrightarrow} M = \mathcal{L}_{\mathcal{A}}(q).$$

Нека дефинираме $f' : Q' \to Q^{\mathcal{B}}$ като

$$f'([q]_{\equiv_{\mathcal{A}}}) \overset{\text{деф}}{=} f(q).$$

От *Твърдение 2.22* имаме, че f' е биекция. Заключаваме, че $|Q'| = |Q^{\mathcal{B}}|$. □

2.10.1 Кубичен алгоритъм за минимизация

При даден език L и детерминиран краен автомат $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$, който го разпознава, целта ни е да построим нов детерминиран краен автомат $\mathcal{A}'$, който има толкова състояния колкото са класовете на еквивалентност на релацията $\equiv_{\mathcal{A}}$. Това ще направим като „слеем" състоянията на $\mathcal{A}$, които са еквивалентни относно релацията $\equiv_{\mathcal{A}}$. Проблемът с намирането на класовете на еквивалентност на релацията $\equiv_{\mathcal{A}}$ е кванторът $(\forall \omega \in \Sigma^\star)$ във нейната дефиниция (чрез Формула 2.10), защото $\Sigma^\star$ е безкрайно множество от думи. За да разрешим този проблем, ще разгледаме *апроксимации* на езиците $\mathcal{L}_{\mathcal{A}}(q)$. За естествено число n, да означим

$$\mathcal{L}^n_{\mathcal{A}}(p) \overset{\text{деф}}{=} \{\omega \in \Sigma^\star \mid |\omega| \leq n \;\&\; \delta^\star(p, \omega) \in F\}.$$

Можем ли да дадем горна граница m, така че $L(\mathcal{A}) = \bigcup_{n \leq m} \mathcal{L}^n_{\mathcal{A}}(q_{\texttt{start}})$?

Лесно се съобразява, че

$$L(\mathcal{A}) = \bigcup_{n \geq 0} \mathcal{L}^n_{\mathcal{A}}(q_{\texttt{start}}).$$

За всяко естествено число n, дефинираме бинарните релации $\equiv^n_{\mathcal{A}}$ върху Q по следния начин:

$$p \equiv^n_{\mathcal{A}} q \overset{\text{деф}}{\Leftrightarrow} \mathcal{L}^n_{\mathcal{A}}(p) = \mathcal{L}^n_{\mathcal{A}}(q).$$

Релациите $\equiv^n_{\mathcal{A}}$ представляват апроксимации на релацията $\equiv_{\mathcal{A}}$. Обърнете внимание, че за всяко n, $\equiv^n_{\mathcal{A}}$ е *по-груба* релация от $\equiv^{n+1}_{\mathcal{A}}$, която на свой ред е по-груба от $\equiv_{\mathcal{A}}$. Алгоритъмът строи $\equiv^n_{\mathcal{A}}$ докато не срещнем n, за което

$$\equiv^n_{\mathcal{A}} = \equiv^{n+1}_{\mathcal{A}} .$$

Тъй като броят на класовете на еквивалентност на $\equiv_{\mathcal{A}}$ е краен (той е $\leq |Q|$), то със сигурност ще намерим такова n, за което $\equiv^n_{\mathcal{A}} = \equiv^{n+1}_{\mathcal{A}}$. Тогава заключаваме, че за това n имаме, че

$$\equiv_{\mathcal{A}} = \equiv^n_{\mathcal{A}} .$$

Ако $q \in F$, то $\mathcal{L}^0_{\mathcal{A}}(q) = F$ и ако $q \notin F$, то $\mathcal{L}^0_{\mathcal{A}}(q) = Q \setminus F$.

Понеже единствената дума с дължина 0 е ε и по определение $\delta^\star(p, \varepsilon) = p$, лесно се съобразява, че $\equiv^0_{\mathcal{A}}$ има два класа на еквивалентност. Единият е F, а другият е $Q \setminus F$.

Вече имаме базовия случай за $n = 0$. Да видим сега как можем да намерим $\equiv^{n+1}_{\mathcal{A}}$ при положение, че вече сме намерили $\equiv^n_{\mathcal{A}}$.

Твърдение 2.23. За всеки две състояния p и q, и всяко естествено число n, $p \equiv_{\mathcal{A}}^{n+1} q$ точно тогава, когато:

а) $p \equiv_{\mathcal{A}}^{n} q$ и

б) $(\forall a \in \Sigma)[\delta(q,a) \equiv_{\mathcal{A}}^{n} \delta(p,a)]$.

Доказателство. Да положим $\Sigma^{\leq n} \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid |\alpha| \leq n\}$. Получаваме еквивалентностите: [PL98, стр. 99]

$$\begin{aligned}
p \equiv_{\mathcal{A}}^{n+1} q &\Leftrightarrow (\forall \alpha \in \Sigma^{\leq n+1})[\delta^\star(p,\alpha) \in F \Leftrightarrow \delta^\star(q,\alpha) \in F] \\
&\Leftrightarrow (\forall \alpha \in \Sigma^{\leq n})[\delta^\star(p,\alpha) \in F \Leftrightarrow \delta^\star(q,\alpha) \in F] \,\&\, \\
&\quad (\forall a \in \Sigma)(\forall \beta \in \Sigma^{\leq n})[\delta^\star(p,a\beta) \in F \Leftrightarrow \delta^\star(q,a\beta) \in F] \\
&\Leftrightarrow (\forall \alpha \in \Sigma^{\leq n})[\delta^\star(p,\alpha) \in F \Leftrightarrow \delta^\star(q,\alpha) \in F] \,\&\, \\
&\quad (\forall a \in \Sigma)(\forall \beta \in \Sigma^{\leq n})[\delta^\star(\delta(p,a),\beta) \in F \Leftrightarrow \delta^\star(\delta(q,a),\beta) \in F] \,\&\, \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n} q \,\&\, (\forall a \in \Sigma)[\delta(p,a) \equiv_{\mathcal{A}}^{n} \delta(q,a)].
\end{aligned}$$

□

Твърдение 2.24. Нека $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{n+1}$. Тогава:

$$m > n \implies \equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{m} . \tag{2.12}$$

Доказателство. Ще докажем Свойство (2.12) с индукция по m за $m > n$.

- Базата на индукцията е случаят $m = n+1$, за който Свойство (2.12) е изпълнено по условие.
- Индукционното ни предположение е, че $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{m}$ за някое $m > n+1$.
- Индукционната ни стъпка е да докажем, че $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{m+1}$. За произволни състояния p и q имаме следните еквивалентности:

$$\begin{aligned}
p \equiv_{\mathcal{A}}^{m+1} q &\Leftrightarrow p \equiv_{\mathcal{A}}^{m} q \,\&\, (\forall a \in \Sigma)[\delta(p,a) \equiv_{\mathcal{A}}^{m} \delta(q,a)] && \text{// от } \textit{Твърдение 2.23} \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n} q \,\&\, (\forall a \in \Sigma)[\delta(p,a) \equiv_{\mathcal{A}}^{n} \delta(q,a)] && \text{// от И.П.} \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n+1} q && \text{// от } \textit{Твърдение 2.23} \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n} q. && \text{// от условието}
\end{aligned}$$

□

Твърдение 2.25. Докажете, че за произволно естествено число n,

$$\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{n+1} \implies \equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}} .$$

Доказателство. Нека $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{n+1}$. Ще докажем, че за произволни състояния p и q е изпълено, че:

$$p \equiv_{\mathcal{A}} q \Leftrightarrow p \equiv_{\mathcal{A}}^{n} q.$$

Ясно е, че $p \equiv_{\mathcal{A}} q \implies p \equiv_{\mathcal{A}}^{n} q$. Да видим защо имаме и обратната посока, т.е. защо $p \equiv_{\mathcal{A}}^{n} q \implies p \equiv_{\mathcal{A}} q$. Ние ще докажем контрапозицията на импликацията, т.е. ще докажем следното:

$$p \not\equiv_{\mathcal{A}} q \implies p \not\equiv_{\mathcal{A}}^{n} q.$$

И така, нека $p \not\equiv_{\mathcal{A}} q$. Това означава, че съществува дума α, за която:

$$\neg(\delta^\star(p, \alpha) \in F \Leftrightarrow \delta^\star(q, \alpha) \in F).$$

Това означава, че $p \not\equiv_{\mathcal{A}}^{|\alpha|} q$. Имаме два случая.

- Нека $|\alpha| \leq n$. Тогава $p \not\equiv_{\mathcal{A}}^{n} q$, защото от дефиницията следва, че

$$p \not\equiv_{\mathcal{A}}^{|\alpha|} q \implies p \not\equiv_{\mathcal{A}}^{n} q.$$

- Ако $|\alpha| > n$, от *Твърдение 2.24* имаме, че

$$p \equiv_{\mathcal{A}}^{n} q \implies p \equiv_{\mathcal{A}}^{|\alpha|} q,$$

чиято контрапозиция е:

$$p \not\equiv_{\mathcal{A}}^{|\alpha|} q \implies p \not\equiv_{\mathcal{A}}^{n} q.$$

Заключаваме, че $p \not\equiv_{\mathcal{A}}^{n} q$.

□

Твърдение 2.26. За всеки ДКА $\mathcal{A}$ е изпълнено, че $\equiv_{\mathcal{A}}^{|Q|-1} = \equiv_{\mathcal{A}}$.

Доказателство. Имаме два случая, които трябва да разгледаме.

- $\equiv_{\mathcal{A}}^{0}$ има точно два класа на еквивалентност.
- $\equiv_{\mathcal{A}}^{1}$ има не повече от три класа на еквивалентност
- $\equiv_{\mathcal{A}}^{|Q|-2}$ има не повече от $|Q|$ класа на еквивалентност.

- Ако съществува $n < |Q| - 2$, за което $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{n+1}$, то според *Твърдение 2.24* и *Твърдение 2.25* получаваме, че $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{|Q|-1} = \equiv_{\mathcal{A}}$.
- Нека сега приемем, че за всяко $n < |Q| - 2$ е изпълнено, че $\equiv_{\mathcal{A}}^{n} \neq \equiv_{\mathcal{A}}^{n+1}$. Това означава, че всеки клас на еквивалентност на $\equiv_{\mathcal{A}}^{|Q|-2}$ съдържа точно едно състояние, защото всеки клас на еквивалентност съдържа поне едно състояние, а ние имаме точно $|Q|$ на брой състояния и поне $|Q|$ на брой класове на еквивалентност. Тогава със сигурност имаме, че $\equiv_{\mathcal{A}}^{|Q|-2} = \equiv_{\mathcal{A}}^{|Q|-1}$, защото

$$p \equiv_{\mathcal{A}}^{|Q|-1} q \implies p \equiv_{\mathcal{A}}^{|Q|-2} q$$

и няма как $\equiv_{\mathcal{A}}^{|Q|-1}$ да „раздроби" някой клас на $\equiv_{\mathcal{A}}^{|Q|-2}$. Тогава от *Твърдение 2.25* следва, че $\equiv_{\mathcal{A}}^{|Q|-1} = \equiv_{\mathcal{A}}$.

□

[Sha08, стр. 83]

Algorithm 1 Кубичен алгоритъм за минимизация

```
1:  for all p < |Q| do                          ▷ състоянията са индексирани от 0 до |Q| − 1
2:      for all q < |Q| do
3:          if (p ∈ F ⇔ q ∉ F) then
4:              E[p][q] = false                 ▷ Имаме, че p ≢⁰_A q
5:          else
6:              E[p][q] = true                  ▷ Имаме, че p ≡⁰_A q
7:  repeat
8:      ready = true
9:      for all p < |Q| do
10:         for all q < |Q| do
11:             if E[p][q] then
12:                 for all a ∈ Σ do            ▷ Прилагаме Твърдение 2.23
13:                     if ! E[δ(p,a)][δ(q,a)] then   ▷ p ≡ⁿ_A q, но δ(p,a) ≢ⁿ_A δ(q,a)
14:                         E[p][q] = false     ▷ Тогава p ≢^{n+1}_A q
15:                         ready = false       ▷ Имаме разцепване на клас
16: until ready                                 ▷ Ясно е, че няма да зациклим
```

Задача 2.28. Докажете, че за всеки две състояния p и q,

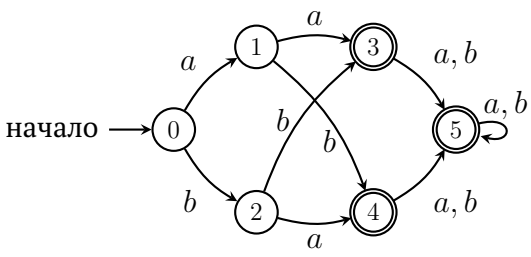
$$p \equiv_{\mathcal{A}} q \;\Leftrightarrow\; \texttt{E[p][q] == true}.$$

Задача 2.29. Съобразете, че *Алгоритъм (1)* има времева сложност $\mathcal{O}(|\Sigma| \cdot |Q|^3)$.

Примерни задачи

[Koz97, стр. 79]

Задача 2.30. Постройте минимален автомат $\mathcal{A}'$ разпознаващ езика на детерминирания краен автомат $\mathcal{A}$.



Фигура 2.25: Автоматът $\mathcal{A}$.

В процедурата, която следваме тук, изобщо не се интересуваме какъв е езика на автомата $\mathcal{A}$.
✍ Все пак съобразете, че езикът на автомата $\mathcal{A}$ е $\{\omega \in \{a,b\}^\star \mid |\omega| \geq 2\}$.

Решение. Ще приложим алгоритъма за минимизация за да получим минималния автомат за езика L. За всяко $n = 0, 1, 2, \ldots$, ще намерим класовете на еквивалентност на $\equiv^n_{\mathcal{A}}$, докато не намерим n, за което $\equiv^n_{\mathcal{A}} = \equiv^{n+1}_{\mathcal{A}}$. За първото такова n, според *Твърдение 2.25*, знаем, че $\equiv^n_{\mathcal{A}} = \equiv_{\mathcal{A}}$.

- Класовете на еквивалентност на $\equiv^0_{\mathcal{A}}$ са два. Те са
$$A_0 = Q \setminus F = \{0, 1, 2\} \text{ и}$$
$$A_1 = F = \{3, 4, 5\}.$$

- Сега да видим дали можем да разбием някои от класовете на еквивалентност на $\equiv^0_{\mathcal{A}}$.

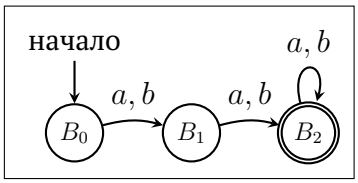
$\equiv^0_{\mathcal{A}}$	A_0			A_1		
Q	0	1	2	$3^\star$	$4^\star$	$5^\star$
a	A_0	A_1	A_1	A_1	A_1	A_1
b	A_0	A_1	A_1	A_1	A_1	A_1

Като използваме *Твърдение 2.23*, виждаме, че $0 \not\equiv^1_{\mathcal{A}} 1$, защото $\delta(0,a) \not\equiv^0_{\mathcal{A}} \delta(1,a)$. От друга страна, $1 \equiv^1_{\mathcal{A}} 2$. Класовете на еквивалентност на $\equiv^1_{\mathcal{A}}$ са следните множества:
$$B_0 = \{0\},$$
$$B_1 = \{1, 2\},$$
$$B_2 = \{3, 4, 5\}.$$

- Сега да видим дали можем да разбием някои от класовете на еквивалентност на $\equiv^1_{\mathcal{A}}$.

$\equiv^1_{\mathcal{A}}$	B_0	B_1		B_2		
Q	0	1	2	$3^\star$	$4^\star$	$5^\star$
a	B_1	B_2	B_2	B_2	B_2	B_2
b	B_1	B_2	B_2	B_2	B_2	B_2

Виждаме, че $\equiv^1_{\mathcal{A}} = \equiv^2_{\mathcal{A}}$, което означава, че $\equiv_{\mathcal{A}} = \equiv^1_{\mathcal{A}}$. Следователно, минималният автомат $\mathcal{M}$ има три състояния.



Фигура 2.26: Минималният автомат $\mathcal{A}'$ за $\mathcal{L}(\mathcal{A})$.

Минималният автомат $\mathcal{A}'$ може да се представи и таблично като опишем действието на функцията на преходите δ' по следния начин:

δ'	B_0	B_1	B_2
a	B_1	B_2	B_2
b	B_1	B_2	B_2

□

Забележка. За този пример можем директно да съобразим кои състояния можем да „слеем" за да получим минималния автомат. Първо, очевидно е, че $\mathcal{L}_{\mathcal{A}}(5) = \{a,b\}^\star$. Тогава лесно се вижда, че

$$\begin{aligned}\mathcal{L}_{\mathcal{A}}(3) &= \{\varepsilon\} \cup \{a,b\} \cdot \mathcal{L}_{\mathcal{A}}(5)\\ &= \{\varepsilon\} \cup \{a,b\} \cdot \{a,b\}^\star\\ &= \{a,b\}^\star\\ \mathcal{L}_{\mathcal{A}}(4) &= \{\varepsilon\} \cup \{a,b\} \cdot \mathcal{L}_{\mathcal{A}}(5)\\ &= \{a,b\}^\star.\end{aligned}$$

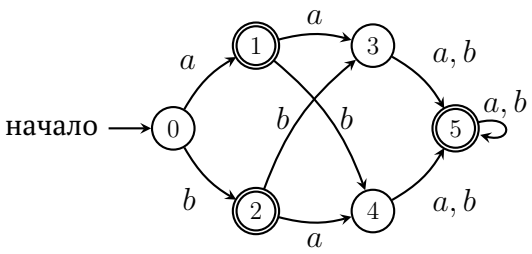
Така получаваме, че $\mathcal{L}_{\mathcal{A}}(3) = \mathcal{L}_{\mathcal{A}}(4) = \mathcal{L}_{\mathcal{A}}(5)$.

Сега, щом $\mathcal{L}_{\mathcal{A}}(3) = \mathcal{L}_{\mathcal{A}}(4)$, то имаме и че $\mathcal{L}_{\mathcal{A}}(1) = \mathcal{L}_{\mathcal{A}}(2)$, защото

$$\begin{aligned}\mathcal{L}_{\mathcal{A}}(1) &= \{a\} \cdot \mathcal{L}_{\mathcal{A}}(3) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(4)\\ \mathcal{L}_{\mathcal{A}}(2) &= \{a\} \cdot \mathcal{L}_{\mathcal{A}}(4) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(3).\end{aligned}$$

Естествено, при по-сложни примери би ни било трудно да съобразим кои състояния можем да „слеем", защото проверката дали два регулярни израза описват един и същ език не е лека задача.

Задача 2.31. Постройте минимален автомат $\mathcal{A}'$ разпознаващ $\mathcal{L}(\mathcal{A})$.



Фигура 2.27: Автоматът $\mathcal{A}$.

Решение. Отново следваме същата процедура за минимизация. Ще намерим класовете на еквивалентност на $\equiv^n_{\mathcal{A}}$, докато не намерим n, за което $\equiv^n_{\mathcal{A}} = \equiv^{n+1}_{\mathcal{A}}$.

- Класовете на екиваленост на $\equiv^0_{\mathcal{A}}$ са следните:

$$A_0 \stackrel{\text{деф}}{=} Q \setminus F = \{0, 3, 4\} \text{ и}$$
$$A_1 \stackrel{\text{деф}}{=} F = \{1, 2, 5\}.$$

- Разбиваме класовете на еквивалентност на $\equiv^0_{\mathcal{A}}$ като използваме *Твърдение 2.23*.

$\equiv^0_{\mathcal{A}}$	A_0			A_1		
Q	0	3	4	$1^\star$	$2^\star$	$5^\star$
a	A_1	A_1	A_1	A_0	A_0	A_1
b	A_1	A_1	A_1	A_0	A_0	A_1

Виждаме, че $1 \not\equiv^1_{\mathcal{A}} 5$ и $1 \equiv^0_{\mathcal{A}} 5$. Следователно, $\equiv^0_{\mathcal{A}} \neq \equiv^1_{\mathcal{A}}$. Класовете на еквивалентност на $\equiv^1_{\mathcal{A}}$ са следните:

$$B_0 \stackrel{\text{деф}}{=} \{0, 3, 4\},$$
$$B_1 \stackrel{\text{деф}}{=} \{1, 2\},$$
$$B_2 \stackrel{\text{деф}}{=} \{5\}.$$

- Сега се опитваме да разбием класовете на еквивалентност на $\equiv^1_{\mathcal{A}}$.

$\equiv^1_{\mathcal{A}}$	B_0			B_1		B_2
Q	0	3	4	$1^\star$	$2^\star$	$5^\star$
a	B_1	B_2	B_2	B_0	B_0	B_2
b	B_1	B_2	B_2	B_0	B_0	B_2

Имаме, че $0 \equiv^1_{\mathcal{A}} 3$, но $0 \not\equiv^2_{\mathcal{A}} 3$. Следователно $\equiv^1_{\mathcal{A}} \neq \equiv^2_{\mathcal{A}}$. Класовете на еквивалентност на $\equiv^2_{\mathcal{A}}$ са следните:

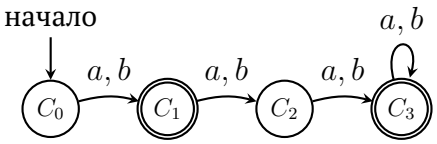
$$C_0 \stackrel{\text{деф}}{=} \{0\},$$
$$C_1 \stackrel{\text{деф}}{=} \{1, 2\},$$
$$C_2 \stackrel{\text{деф}}{=} \{3, 4\},$$
$$C_3 \stackrel{\text{деф}}{=} \{5\}.$$

- Отново опитваме да разбием класовете на релацията $\equiv^2_{\mathcal{A}}$.

$\equiv^2_{\mathcal{A}}$	C_0	C_1		C_2		C_3
Q	0	$1^\star$	$2^\star$	3	4	$5^\star$
a	C_1	C_2	C_2	C_3	C_3	C_3
b	C_1	C_2	C_2	C_3	C_3	C_3

Виждаме, че не можем да разбием C_1 или C_2. Следователно, $\equiv^2_{\mathcal{A}} = \equiv^3_{\mathcal{A}}$. Оттук следва, че $\equiv^2_{\mathcal{A}} = \equiv_{\mathcal{A}}$ и минималният автомат разпознаващ езика L има четири състояния. Вижте *Фигура 2.28* за преходите на минималния автомат, които могат да се представят и табличо чрез функцията на преходите:

δ	C_0	C_1	C_2	C_3
a	C_1	C_2	C_3	C_3
b	C_1	C_2	C_3	C_3



Фигура 2.28: Получаваме минималния автомат $\mathcal{A}'$ за езика на $\mathcal{A}$.

□

Забележка. Както в предишния пример, тук също може директно да се съобрази, че $\mathcal{L}_{\mathcal{A}}(3) = \mathcal{L}_{\mathcal{A}}(4)$, както и $\mathcal{L}_{\mathcal{A}}(1) = \mathcal{L}_{\mathcal{A}}(2)$.

✍ Съобразете, че езикът на автомата $\mathcal{A}$ е $\{\omega \in \{a, b\}^\star \mid |\omega| \neq 0, 2\}$.

2.11 Допълнителни задачи

2.11.1 Лесни задачи

$|\omega|_a \stackrel{\text{деф}}{=}$ броят на срещанията на буквата a в думата ω,
$|\omega| \stackrel{\text{деф}}{=}$ дължината на ω.

Задача 2.32. За всеки от следните езици L, постройте минимален краен детерминиран автомат $\mathcal{A}$, който разпознава езика L, където:

а) $L = \{a^n b \mid n \geq 0\}$;

б) $L = \{a, b\}^\star \setminus \{\varepsilon\}$;

в) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a$ и $|\omega|_b$ са четни$\}$;

г) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a$ е четно $\&\ |\omega|_b$ е нечетно$\}$;

д) $L = \{a^n b^m \mid n, m \geq 0\}$;

е) $L = \{a^n b^m \mid n, m \geq 1\}$;

ж) $L = \{a, b\}^\star \setminus \{a\}$;

з) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a \geq 2 \ \vee\ |\omega|_b \leq 3\}$;

и) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a \geq 2 \ \&\ |\omega|_b \geq 1\}$;

к) $L = \{\omega \in \{a, b\}^\star \mid$ на всяка нечетна позиция на ω е буквата $a\}$;

л) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a$ е четно $\&\ |\omega|_b \leq 1\}$;

м) $L = \{\omega \in \{a, b\}^\star \mid |\omega| \leq 3\}$;

н) $L = \{\omega \in \{a, b\}^\star \mid \omega$ не започва с $ab\}$;

о) $L = \{\omega \in \{a, b\}^\star \mid \omega$ завършва с ab или ba$\}$;

п) $L = \{\omega \in \{a, b\}^\star \mid \omega$ започва или завършва с $a\}$;

р) $L = \{\omega \in \{a, b\}^\star \mid \omega$ започва с $a \ \Leftrightarrow\ \omega$ завършва с $b\}$;

с) $L = \{\omega \in \{a, b\}^\star \mid |\omega| \equiv 0 \pmod 2 \ \&\ |\omega|_a = 1\}$;

т) $L = \{\omega \in \{a, b\}^\star \mid$ всяко a в ω се следва веднага от поне едно $b\}$;

у) $L = \{\omega \in \{a, b\}^\star \mid |\omega| \equiv 0 \bmod 3\}$;

ф) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a \equiv 1 \bmod 3\}$;

х) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a \equiv 0 \bmod 3 \ \&\ |\omega|_b \equiv 1 \bmod 2\}$;

ц) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a \equiv 0 \bmod 2 \ \vee\ |\omega|_b = 2\}$;

ч) $L = \{\omega \in \{a, b\}^\star \mid \omega$ съдържа равен брой срещания на ab и на $ba\}$;

ш) $L = \{\omega_1 \sharp \omega_2 \sharp \omega_3 \mid |\omega_1| \geq 2 \ \&\ |\omega_2| \geq 3 \ \&\ |\omega_3| \geq 4 \ \&\ \omega_i \in \{a, b\}^\star$ за $i = 1, 2, 3\}$.

Задача 2.33. Нека $\Sigma = \{a, b\}$. Проверете дали L е регулярен, където

а) $L = \{a^i b^i \mid i \in \mathbb{N}\}$;

б) $L = \{a^i b^i \mid i, j \in \mathbb{N} \ \&\ i \neq j\}$;

в) $L = \{a^i b^j \mid i > j\}$;

г) $L = \{a^n b^m \mid n$ дели $m\}$.

д) $L = \{a^{2n} \mid n \geq 1\}$;

е) $L = \{a^m b^n a^{m+n} \mid m \geq 1 \ \&\ n \geq 1\}$;

ж) $L = \{a^{n.m} \mid n, m$ са прости числа$\}$;

з) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a = |\omega|_b\}$;

и) $L = \{\omega\omega \mid \omega \in \{a, b\}^\star\}$;

к) $L = \{\omega\omega^{\mathrm{rev}} \mid \omega \in \{a, b\}^\star\}$;

л) $L = \{\alpha\beta\beta \in \{a, b\}^\star \mid \beta \neq \varepsilon\}$;

м) $L = \{a^n b^n c^n \mid n \geq 0\}$;

н) $L = \{\omega\omega\omega \mid \omega \in \Sigma^\star\}$;

о) $L = \{a^{2^n} \mid n \geq 0\}$;

п) $L = \{a^m b^n \mid n \neq m\}$;

р) $L = \{a^{n!} b^{n!} \mid n \neq 1\}$;

с) $L = \{a^{f_n} \mid f_0 = f_1 = 1 \,\&\, f_{n+2} = f_{n+1} + f_n\}$;

т) $L = \{\alpha \in \Sigma^\star \mid |\,|\alpha|_a - |\alpha|_b\,| \leq 2\}$;

у) $L = \{\alpha\beta\alpha \mid \alpha, \beta \in \Sigma^\star \,\&\, |\beta| \leq |\alpha|\}$;

ф) $L = \{\beta\gamma\gamma^{\mathtt{rev}} \mid \beta, \gamma \in \Sigma^\star \,\&\, |\beta| \leq |\gamma|\}$;

х) $L = \{c^k a^n b^m \mid k, m, n > 0 \,\&\, n \neq m\}$;

ц) $L = \{c^k a^n b^n \mid k > 0 \,\&\, n \geq 0\} \cup \{a, b\}^\star$;

ч) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a$ не дели $|\omega|_b\}$;

ш) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a < |\omega|_b\}$;

щ) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a = 2|\omega|_b\}$;

ю) $L = \{\omega \in \{a, b\}^\star \mid |\,|\omega|_a - |\omega|_b\,| \leq 3\}$.

Задача 2.34. Докажете, че следните езици са регулярни:

а) $L = \{\alpha \in \{a, b\}^\star \mid$ за всяка представка ω на α имаме $|\,|\omega|_a - |\omega|_b\,| \leq 2\}$;

б) $L = \{\alpha \in \{a, b\}^\star \mid$ за някоя представка ω на α имаме $|\,|\omega|_a - |\omega|_b\,| > 2\}$;

в) $L = \{\alpha \in \{a, b\}^\star \mid$ за някоя наставка ω на α имаме $|\,|\omega|_a - |\omega|_b\,| > 2\}$.

Задача 2.35. Нека $\Sigma = \{a, b, c, d\}$. Да се докаже, че езикът

$$L = \{a_1 a_2 \cdots a_{2n} \in \Sigma^\star \mid (\forall j \in [1, n])[a_{2j-1} = a_{2j}] \,\&\, d \text{ се среща} \leq 3 \text{ пъти}\}$$

е регулярен.

Задача 2.36. Нека L_1 и L_2 са регулярни езици. Докажете, че L също е регулярен език, където

$$L = \{\alpha \mid (\exists \beta, \gamma)[\beta\alpha\gamma \in L_1] \,\&\, \alpha \in L_2 \vee \alpha^{\mathtt{rev}} \in L_2\}.$$

Определение 2.3. Да фиксираме две азбуки Σ_1 и Σ_2. Хомоморфизъм е изображение $h : \Sigma_1^\star \to \Sigma_2^\star$ със свойството, че за всеки две думи $\alpha, \beta \in \Sigma_1^\star$,

$$h(\alpha\beta) = h(\alpha) \cdot h(\beta).$$

Лесно се съобразява, че за всеки хомоморфизъм h, $h(\varepsilon) = \varepsilon$.

Задача 2.37. Нека $L \subseteq \Sigma_1^\star$ е регулярен език и $h : \Sigma_1^\star \to \Sigma_2^\star$ е хомоморфизъм. Тогава $h(L) = \{h(\alpha) \in \Sigma_2^\star \mid \alpha \in L\}$ е регулярен.

Упътване. Индукция по построението на регулярни езици. □

Задача 2.38. Нека $L \subseteq \Sigma_2^\star$ е регулярен език и $h : \Sigma_1^\star \to \Sigma_2^\star$ е хомоморфизъм. Тогава езикът $h^{-1}(L) = \{\alpha \in \Sigma_1^\star \mid h(\alpha) \in L\}$ е регулярен.

Упътване. Конструкция на автомат за $h^{-1}(L)$ при даден автомат за L. □

Задача 2.39. Нека Σ_1 и Σ_2 са непресичащи се азбуки, а L_1 и L_2 са езици съответно над Σ_1 и Σ_2. За една дума $\omega \in (\Sigma_1 \cup \Sigma_2)^\star$, нека с $\omega_i \in \Sigma_i^\star$ да означим редицата от букви от Σ_i в реда, в който се срещат в ω. Да разгледаме следния език

$$L_1 \oplus L_2 = \{\omega \in (\Sigma_1 \cup \Sigma_2)^\star \mid \omega_1 \in L_1 \,\&\, \omega_2 \in L_2 \,\&\, |\omega_1| = |\omega_2|\}.$$

Да а) Вярно ли е, че ако L_1 е краен, то $L_1 \oplus L_2$ е регулярен език?

Не б) Вярно ли е, че ако L_1 и L_2 са регулярни езици, то $L_1 \oplus L_2$ е регулярен език?

Задача 2.40. Нека Σ_1 и Σ_2 са непресичащи се азбуки, а L_1 и L_2 са езици съответно над Σ_1 и Σ_2. За една дума $\omega \in (\Sigma_1 \cup \Sigma_2)^\star$, нека с $\omega_i \in \Sigma_i^\star$ да означим редицата от букви от Σ_i в реда, в който се срещат в ω. Да разгледаме следния език

$$L_1 \oplus L_2 = \{\omega \in (\Sigma_1 \cup \Sigma_2)^\star \mid \omega_1 \in L_1 \,\&\, \omega_2 \in L_2\}.$$

Да Вярно ли е, че ако L_1 и L_2 са регулярни езици, то $L_1 \oplus L_2$ е регулярен език?

2.11.2 Не толкова лесни задачи

Тази задача е давана на държавен изпит във ФМИ през септември 2021 год.

Задача 2.41. Нека L е регулярен език над азбуката Σ. Вярно ли е, че езикът $L' = \{\beta\alpha \in \Sigma^\star \mid \alpha\beta \in L\}$ е регулярен? Обосновете отговора си като приложите доказателство!

[PL98, стр. 84]

Задача 2.42. При дадени езици L и M над азбуката Σ, да разгледаме:

а) $\texttt{Pref}(L) = \{\alpha \in \Sigma^\star \mid (\exists\beta \in \Sigma^\star)[\alpha\beta \in L]\}$;

б) $\texttt{NoPref}(L) = \{\alpha \in L \mid$ не съществува префикс на α в $L\}$;

в) $\texttt{NoExtend}(L) = \{\alpha \in L \mid \alpha$ не е префикс на никоя дума от $L\}$;

г) $\text{Suf}(L) = \{\beta \in \Sigma^\star \mid (\exists\alpha \in \Sigma^\star)[\alpha\beta \in L]\}$;

д) $\text{Infix}(L) = \{\alpha \mid (\exists\beta,\gamma \in \Sigma^\star)[\beta\alpha\gamma \in L]\}$;

е) $\frac{1}{2}(L) = \{\omega \in \Sigma^\star \mid (\exists\alpha \in \Sigma^\star)[\omega\alpha \in L \,\&\, |\omega| = |\alpha|]\}$;

right quotient of L by ω

ж) $L/_\omega = \{\alpha \in \Sigma^\star \mid \alpha\omega \in L\}$;

з) $L/_M = \{\alpha \in \Sigma^\star \mid (\exists\beta \in M)[\alpha\beta \in L]\}$;

right quotient of L by M

и) $L^{-1}(M) = \{\beta \mid (\exists\alpha \in L)[\alpha\beta \in M]\}$;

к) $\text{Max}(L) = \{\alpha \in \Sigma^\star \mid (\forall\beta \in \Sigma^\star)[\beta \neq \varepsilon \implies \alpha\beta \notin L]\}$.

За всички тези езици, докажете, че са регулярни при условие, че L и M са регулярни. Освен това, докажете, че $L/_M$ е регулярен и при условието, че L е регулярен, но M е произволен език над азбуката Σ.

Тази конструкция няма да бъде ефективна

Упътване.

а) Индукция по дефиницията на регулярен израз.

в) Най-лесно е да се построи автомат за $\text{Infix}(L)$ като се използва автомата за L.

г) Конструкция с автомат за L и автомат за $L^{\texttt{rev}}$.

□

Задача 2.43. Да фиксираме азбука само с един символ $\Sigma = \{a\}$. Да положим за всяко $p, q \in \mathbb{N}$, [Koz97, стр. 75]; [PL98, стр. 89]

$$L(p,q) = \{a^k \mid (\exists n \in \mathbb{N})[k = p + q \cdot n]\}.$$

Ако за един език L съществуват константи $p_1, \ldots, p_k$ и $q_1, \ldots, q_k$, такива че

$$L = \bigcup_{1 \leq i \leq k} L(p_i, q_i),$$

то казваме, че L е *породен от аритметични прогресии*.

а) Докажете, че $L \subseteq \{a\}^\star$ е регулярен език точно тогава, когато L е породен от аритметична прогресия.

б) За произволна азбука Σ, докажете, че ако $L \subseteq \Sigma^\star$ е регулярен език, то езикът $\{a^{|\omega|} \mid \omega \in L\}$ е породен от аритметична прогресия.

Упътване.

а) За едната посока, разгледайте ДКА за L.

б) За втората част, разгледайте $h : \Sigma \to \{a\}$ деф. като $(\forall b \in \Sigma)[h(b) = a]$. Докажете, че h е поражда хомоморфизъм между $\Sigma^\star$ и $\{a\}^\star$. Тогава $h(L) = \{a^{|\omega|} \mid \omega \in L\}$, а ние знаем, че регулярните езици са затворени относно хомоморфни образи.

□

Задача 2.44. Вярно ли е, че:

- $\{a^m \mid a^{m^2} \in L(p,q)\}$ е регулярен език ?
- $\{a^m \mid a^{2^m} \in L(p,q)\}$ е регулярен език ?

Задача 2.45. За даден език L над азбуката Σ, да разгледаме езиците:

а) $L' = \{\alpha \in \Sigma^\star \mid (\exists \beta \in \Sigma^\star)[|\alpha| = 2|\beta| \,\&\, \alpha\beta \in L]\}$;

б) $L'' = \{\alpha \in \Sigma^\star \mid (\exists \beta \in \Sigma^\star)[2|\alpha| = |\beta| \,\&\, \alpha\beta \in L]\}$;

в) $\frac{1}{3}(L) = \{\alpha \in \Sigma^\star \mid (\exists \beta, \gamma \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \,\&\, \alpha\beta\gamma \in L]\}$;

г) $\frac{2}{3}(L) = \{\beta \in \Sigma^\star \mid (\exists \beta, \gamma \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \,\&\, \alpha\beta\gamma \in L]\}$;

д) $\frac{3}{3}(L) = \{\gamma \in \Sigma^\star \mid (\exists \beta, \gamma \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \,\&\, \alpha\beta\gamma \in L]\}$;

е) $\hat{L} = \{\alpha\gamma \in \Sigma^\star \mid (\exists \beta, \gamma \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \,\&\, \alpha\beta\gamma \in L]\}$;

ж) $\sqrt{L} = \{\alpha \mid (\exists \beta \in \Sigma^\star)[|\beta| = |\alpha|^2 \,\&\, \alpha\beta \in L]\}$;

з) $\log(L) = \{\alpha \mid (\exists \beta \in \Sigma^\star)[|\beta| = 2^{|\alpha|} \,\&\, \alpha\beta \in L]\}$;

Проверете ако L е регулярен, то кои от горните езици също са регулярни.

Задача 2.46. Да разгледаме езика

$$L = \{\omega \in \{0,1\}^\star \mid \omega \text{ съдържа равен брой поднизове } 01 \text{ и } 10\}.$$

Например, $101 \in L$, защото съдържа по веднъж 10 и 01. $1010 \notin L$, защото съдържа два пъти 10 и само веднъж 01. Докажете, че L е регулярен.

Задача 2.47. Нека L е регулярен език над азбуката $\{a, b\}$. Докажете, че следните езици са регулярни:

а) $\texttt{Diff}_1(L) \stackrel{\text{деф}}{=} \{\alpha \in L \mid (\exists \beta \in L)[|\alpha| = |\beta| \,\&\, \alpha$ се различава от β в една позиция$]\}$;

б) $\texttt{Diff}_n(L) \stackrel{\text{деф}}{=} \{\alpha \in L \mid (\exists \beta \in L)[n \leq |\alpha| = |\beta| \,\&\, \alpha$ се различава от β в n позиции$]\}$;

в) $\texttt{Diff}(L) \stackrel{\text{деф}}{=} \{\alpha \in L \mid (\exists \beta \in L)[|\alpha| = |\beta| \,\&\, \alpha$ се различава от β във всяка позиция$]\}$;

Упътване. Ако $L = \mathcal{L}(\mathcal{A})$, то правим декартово произведение на $\mathcal{A}$ плюс флаг дали сме направили грешка.

Не е ли по-лесно с индукция по построението на регулярните езици ? □

Да обърнем внимание, че езикът $\{\alpha\sharp\beta\sharp\gamma \mid \overline{\alpha}_{(2)} + \overline{\beta}_{(2)} = \overline{\gamma}_{(2)}\}$ не е регулярен.

Задача 2.48. Да разгледаме азбуката:

$$\Sigma_3 = \left\{ \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\1\\1\end{bmatrix}, \ldots, \begin{bmatrix}1\\1\\1\end{bmatrix} \right\}.$$

Докажете, че

$$L = \left\{ \begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix} \in \Sigma_3^\star \mid \overline{\alpha}_{(2)} + \overline{\beta}_{(2)} = \overline{\gamma}_{(2)} \right\}$$

е автоматен език.

Упътване. Доста по-удобно е да построим автомат $\mathcal{A}$, такъв че $\mathcal{L}(\mathcal{A}) = L^{\texttt{rev}}$. Да започнем с състоянието $q_=$, за което искаме да имаме свойството, че за произволно състояние q,

$$\delta^\star(q, \begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}) = q_= \Leftrightarrow \overline{\alpha^{\texttt{rev}}}_{(2)} + \overline{\beta^{\texttt{rev}}}_{(2)} = \overline{\gamma^{\texttt{rev}}}_{(2)}.$$

Понеже за $\overline{\varepsilon}_{(2)} + \overline{\varepsilon}_{(2)} = \overline{\varepsilon}_{(2)}$, състоянието $q_=$ ще бъде начално и финално за $\mathcal{A}$.

Нека $\overline{\alpha}_{(2)} + \overline{\beta}_{(2)} = \overline{\gamma}_{(2)}$. Тогава:

$$\delta(q_=, \begin{bmatrix}0\\0\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_= \qquad // \ \overline{0\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{0\gamma}_{(2)}$$

$$\delta(q_=, \begin{bmatrix}0\\1\\1\end{bmatrix}) \stackrel{\text{деф}}{=} q_= \qquad // \ \overline{0\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{1\gamma}_{(2)}$$

$$\delta(q_=, \begin{bmatrix}1\\0\\1\end{bmatrix}) = q_= \qquad // \ \overline{1\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{1\gamma}_{(2)}$$

Остана случая $\overline{1\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{10\gamma}_{(2)}$. Този случай е по-специален и трябва да бъде разгледан отделно. Трябва да отидем в състояние q_1, в което ще помним, че третия ред трябва да започва с 1-ца. Затова имаме следния преход:

$$\delta(q_=, \begin{bmatrix}1\\1\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_1.$$

За останалите $\gamma \in \Sigma_3$ имаме, че

$$\delta(q_=, \gamma) \stackrel{\text{деф}}{=} q_{\texttt{err}},$$

където $q_{\texttt{err}}$ е състоянието, от което не можем да излезем.

Така трябва да дефинираме функцията на преходите, че за състоянието q_1 трябва да е изпълнено, че за произволно q,

$$\delta^\star(q, \begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}) = q_1 \Leftrightarrow \overline{\alpha^{\texttt{rev}}}_{(2)} + \overline{\beta^{\texttt{rev}}}_{(2)} = \overline{1\gamma^{\texttt{rev}}}_{(2)}.$$

Да разгледаме сега случая $\overline{\alpha}_{(2)} + \overline{\beta}_{(2)} = \overline{1\gamma}_{(2)}$. Тогава:

$$\begin{aligned}
&\delta(q_1, \begin{bmatrix}0\\0\\1\end{bmatrix}) \stackrel{\text{деф}}{=} q_= && // \ \overline{0\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{1\gamma}_{(2)}\\
&\delta(q_1, \begin{bmatrix}1\\1\\1\end{bmatrix}) \stackrel{\text{деф}}{=} q_1 && // \ \overline{1\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{11\gamma}_{(2)}\\
&\delta(q_1, \begin{bmatrix}1\\0\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_1 && // \ \overline{1\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{10\gamma}_{(2)}\\
&\delta(q_1, \begin{bmatrix}0\\1\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_1 && // \ \overline{0\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{10\gamma}_{(2)}\\
&\delta(q_1, \gamma) \stackrel{\text{деф}}{=} q_{\texttt{err}} && // \ \text{за останалите } \gamma \in \Sigma_3
\end{aligned}$$

□

Задача 2.49. Да разгледаме азбуката:

$$\Sigma_2 = \left\{ \begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix} \right\}.$$

Една дума над азбуката Σ_2 ни дава два реда от 0-ли и 1-ци, които ще разглеждаме като числа в двоична бройна система. Да разгледаме езиците:

- $L_1 = \left\{ \begin{bmatrix}\alpha\\\beta\end{bmatrix} \in \Sigma_2^\star \mid \overline{\alpha}_{(2)} < \overline{\beta}_{(2)} \right\}$;
- $L_2 = \left\{ \begin{bmatrix}\alpha\\\beta\end{bmatrix} \in \Sigma_2^\star \mid 3(\overline{\alpha}_{(2)}) = \overline{\beta}_{(2)} \right\}$;
- $L_3 = \left\{ \begin{bmatrix}\alpha\\\beta\end{bmatrix} \in \Sigma_2^\star \mid \alpha = \beta^{\texttt{rev}} \right\}$;

Докажете, че L_1 и L_2 са автоматни, а L_3 не е автоматен.

Упътване. Ще построим автомат $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ за езика $L_1^{\texttt{rev}}$. За улеснение, в рамките на тази задача ще пишем:

- $\alpha \equiv \beta$, ако $\overline{\alpha^{\texttt{rev}}}_{(2)} = \overline{\beta^{\texttt{rev}}}_{(2)}$,
- $\alpha \prec \beta$, ако $\overline{\alpha^{\texttt{rev}}}_{(2)} < \overline{\beta^{\texttt{rev}}}_{(2)}$,
- $\alpha \succ \beta$, ако $\overline{\alpha^{\texttt{rev}}}_{(2)} > \overline{\beta^{\texttt{rev}}}_{(2)}$.

Нека състоянията на автомата са $Q = \{q_=, q_<, q_>\}$. Искаме да е изпълнено свойствата:

- $\delta^\star(q_=, \begin{bmatrix}\alpha\\ \beta\end{bmatrix}) = q_=$ точно тогава, когато $\alpha \equiv \beta$;
- $\delta^\star(q_=, \begin{bmatrix}\alpha\\ \beta\end{bmatrix}) = q_<$ точно тогава, когато $\alpha \prec \beta$;
- $\delta^\star(q_=, \begin{bmatrix}\alpha\\ \beta\end{bmatrix}) = q_>$ точно тогава, когато $\alpha \succ \beta$.

Множеството от финални състояния ще бъде $F = \{q_<\}$, а началното състояние $q_{\texttt{start}} = q_=$. За да дефинираме функцията на преходите, трябва да разгледа няколко случая, в зависимост от това какво е отношението между α и β.

- Нека $\alpha \equiv \beta$. Тогава:
  - $\alpha 0 \equiv \beta 0$ и $\alpha 1 \equiv \beta 1$. Следователно,
    $$\delta(q_=, \begin{bmatrix}0\\0\end{bmatrix}) = \delta(q_=, \begin{bmatrix}1\\1\end{bmatrix}) = q_=.$$
  - $\alpha 0 \prec \beta 1$. Следователно,
    $$\delta(q_=, \begin{bmatrix}0\\1\end{bmatrix}) = q_>.$$
  - $\alpha 1 \succ \beta 0$. Следователно,
    $$\delta(q_=, \begin{bmatrix}1\\0\end{bmatrix}) = q_<.$$
- Нека $\alpha \prec \beta$. Тогава:
  - $\alpha 0 \prec \beta 0$, $\alpha 1 \prec \beta 1$, $\alpha 0 \prec \beta 1$. Следователно,
    $$\delta(q_<, \begin{bmatrix}0\\0\end{bmatrix}) = \delta(q_<, \begin{bmatrix}1\\1\end{bmatrix}) = \delta(q_<, \begin{bmatrix}0\\1\end{bmatrix}) = q_<.$$
  - $\alpha 1 \succ \beta 0$. Следователно,
    $$\delta(q_<, \begin{bmatrix}1\\0\end{bmatrix}) = q_>.$$
- Нека $\alpha \succ \beta$. Тогава:
  - $\alpha 0 \succ \beta 0$, $\alpha 1 \succ \beta 1$, $\alpha 1 \succ \beta 0$. Следователно,
    $$\delta(q_>, \begin{bmatrix}0\\0\end{bmatrix}) = \delta(q_>, \begin{bmatrix}1\\1\end{bmatrix}) = \delta(q_>, \begin{bmatrix}1\\0\end{bmatrix}) = q_>.$$
  - $\alpha 0 \prec \beta 1$. Следователно,
    $$\delta(q_>, \begin{bmatrix}0\\1\end{bmatrix}) = q_<.$$

Докажете, че за така дефинирания автомат $\mathcal{A}$, $\mathcal{L}(\mathcal{A}) = L_1^{\texttt{rev}}$. □

Нека $\Sigma^{\geq k} \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid |\alpha| \geq k\}$.

Задача 2.50. Нека L е регулярен език. Тогава езиците

- $L_1 = \{\alpha \in \Sigma^\star \mid (\exists i)(\exists j)[\, \alpha[i:j] \in L\,]\}$;
- $L_2 = \{\alpha \in \Sigma^\star \mid (\forall i)(\forall j)[\, i < j \implies \alpha[i:j] \in L\,]\}$;
- $L_3 = \{\alpha \in \Sigma^\star \mid (\forall i)(\exists j)[\, i < j \;\&\; \alpha[i:j] \in L\,]\}$;
- $L_4 = \{\alpha \in \Sigma^\star \mid (\exists i)(\forall j)[\, i < j \implies \alpha[i:j] \in L\,]\}$.

също са регулярни.

Упътване. Лесно се вижда, че $L_1 = \Sigma^\star \cdot L \cdot \Sigma^\star$, както и $L_2 = \overline{\Sigma^\star \cdot \overline{L} \cdot \Sigma^\star}$. Да обърнем внимание, че

$$\begin{aligned}\alpha \in \Sigma^\star \cdot L &\Leftrightarrow (\exists \beta \in \Sigma^\star)(\exists \gamma \in L)[\alpha = \beta \cdot \gamma] \\ &\Leftrightarrow (\exists j)[\, \alpha\texttt{[}j : \texttt{]} \in L\,].\end{aligned}$$

Аналогично получаваме, че

$$\begin{aligned}\alpha \in L \cdot \Sigma^\star &\Leftrightarrow (\exists \gamma \in L)(\exists \beta \in \Sigma^\star)[\alpha = \gamma \cdot \beta] \\ &\Leftrightarrow (\exists j)[\, \alpha\texttt{[}: j\texttt{]} \in L\,].\end{aligned}$$

Нека да разгледаме по-подробно следния език:

$$\begin{aligned}\overline{L}_3 &= \{\alpha \in \Sigma^\star \mid (\exists i)(\forall j)[\, i < j \implies \alpha\texttt{[}i : j\texttt{]} \notin L\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\exists i)(\forall j)[\, i < j \implies \alpha\texttt{[}i : j\texttt{]} \in \overline{L}\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\forall j)[\, \alpha\texttt{[}: j\texttt{]} \in \Sigma^\star \cdot \overline{L}\,]\}\end{aligned}$$

Така получаваме, че:

$$\begin{aligned}L_3 &= \{\alpha \in \Sigma^\star \mid (\exists j)[\, \alpha\texttt{[}: j\texttt{]} \notin \Sigma^\star \cdot \overline{L}\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\exists j)[\, \alpha\texttt{[}: j\texttt{]} \in \overline{\Sigma^\star \cdot \overline{L}}\,]\} \\ &= \{\alpha \in \Sigma^\star \mid \alpha \in (\overline{\Sigma^\star \cdot \overline{L}}) \cdot \Sigma^\star\,]\}.\end{aligned}$$

Оттук заключваме, че

$$L_3 = (\overline{\Sigma^\star \cdot \overline{L}}) \cdot \Sigma^\star.$$

Сега лесно можем да съобразим, че

$$L_4 = \overline{\overline{(\Sigma^\star \cdot L)} \cdot \Sigma^\star}.$$

□

Задача 2.51. Нека L е регулярен език над азбуката Σ. За произволно естествено число k, докажете, че езикът

$$L_k = \{\alpha \in \Sigma^\star \mid (\forall i)(\exists j)[\, |\alpha\texttt{[}i : \texttt{]}| \geq k \implies \alpha\texttt{[}i : j\texttt{]} \in L\,]\}$$

е регулярен. С други думи, L_k съдържа тези думи, за които от всяка позиция, с изключение на последните k, започва дума в езика L.

Упътване. Да разпишем по-подробно дефиницията на езика L_k.

$$\begin{aligned}L_k &= \{\alpha \in \Sigma^\star \mid (\forall i)(\exists j)[\, |\alpha\texttt{[}i : \texttt{]}| \geq k \implies \alpha\texttt{[}i : j\texttt{]} \in L\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\forall i)[\, |\alpha\texttt{[}i : \texttt{]}| \geq k \implies (\exists j)[\alpha\texttt{[}i : j\texttt{]} \in L]\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\forall i)[\, \alpha\texttt{[}i : \texttt{]} \in \Sigma^{\geq k} \implies \alpha\texttt{[}i : \texttt{]} \in L \cdot \Sigma^\star\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\forall i)[\, \alpha\texttt{[}i : \texttt{]} \notin \Sigma^{\geq k} \vee \alpha\texttt{[}i : \texttt{]} \in L \cdot \Sigma^\star\,]\} \\ &= \{\alpha \in \Sigma^\star \mid \neg(\exists i)[\, \alpha\texttt{[}i : \texttt{]} \in \Sigma^{\geq k} \,\&\, \alpha\texttt{[}i : \texttt{]} \notin L \cdot \Sigma^\star\,]\}.\end{aligned}$$

Сега е ясно, че:

$$\begin{aligned}\overline{L}_k &= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha\texttt{[}i : \texttt{]} \in \Sigma^{\geq k} \,\&\, \alpha\texttt{[}i : \texttt{]} \notin L \cdot \Sigma^\star\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha\texttt{[}i : \texttt{]} \in \Sigma^{\geq k} \,\&\, \alpha\texttt{[}i : \texttt{]} \in \overline{L \cdot \Sigma^\star}\,]\} \\ &= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha\texttt{[}i : \texttt{]} \in \Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star}\,]\} \\ &= \{\alpha \in \Sigma^\star \mid \alpha \in \Sigma^\star \cdot (\Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star})\} \\ &= \Sigma^\star \cdot (\Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star}).\end{aligned}$$

Тогава

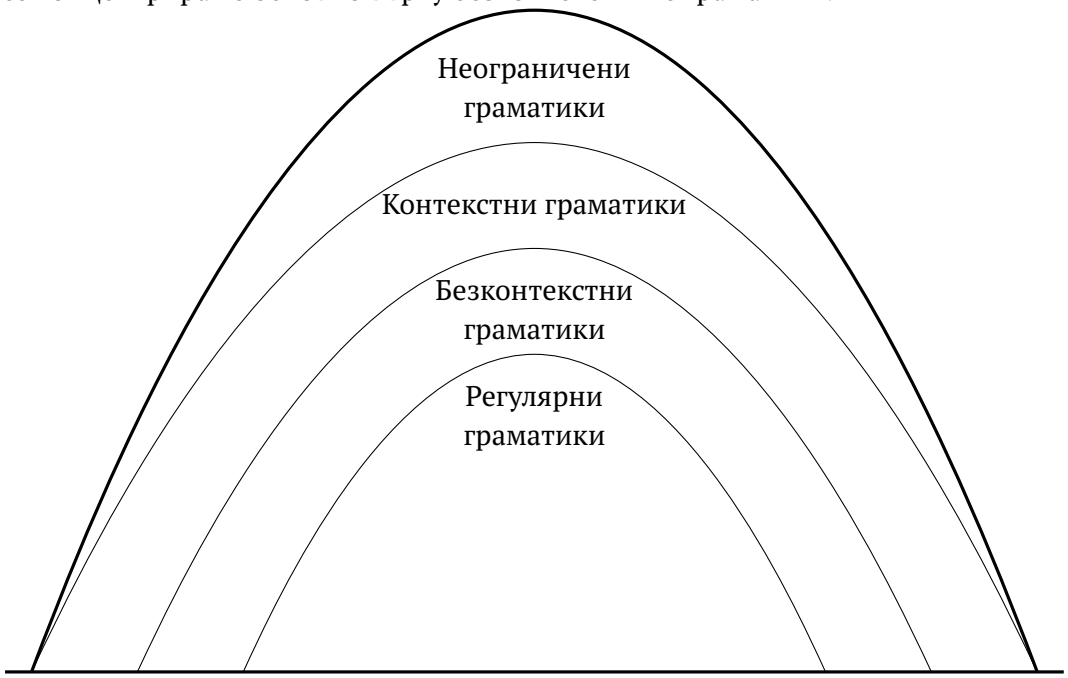
$$L_k = \overline{\Sigma^\star \cdot (\Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star})}.$$

□

Глава 3

Безконтекстни езици и стекови автомати

Ще започнем като разгледаме понятието извод в граматика в най-общия му вид. Граматиките се разделят на няколко вида в зависимост от това какви *ограничения* налагаме върху правилата на граматиката. В следващите няколко глави ще разгледаме различни ограничения. След това ще разгледаме някои класове от граматики като ще се концентрираме основно върху безконтекстните граматики.



3.1 Неограничени граматики

На англ. *unrestricted grammar*. Това са тип 0 граматиките в йерархията на Чомски [HU79, стр. 220].

Неограничена граматика е наредена четворка от вида

$$G = (V, \Sigma, R, S),$$

където:

- V е крайно множество от *променливи* (нетерминали);
- Σ е крайно множество от *букви* (терминали), като $\Sigma \cap V = \emptyset$;
- $R \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)^\star$ е крайно множество от *правила*. За по-добра яснота, обикновено правилата $(\alpha, \beta) \in R$ ще означаваме като $\alpha \to_G \beta$. Когато е ясно за коя граматика говорим, ще пишем просто $\alpha \to \beta$.

В [HU79] правилата се наричат *productions* или *production rules*.

- $S \in V$ е началната променлива (нетерминал).

Ще дефинираме релация $\Rightarrow_G$, която казва, че от думата γ се получава думата γ' чрез прилагане на правилото $\alpha \to_G \beta$ в граматиката G.

Тук отново, когато е ясно за коя граматика говорим, ще пишем просто $\Rightarrow$ вместо $\Rightarrow_G$.

$$\frac{(\alpha, \beta) \in R \qquad \lambda, \rho \in (V \cup \Sigma)^\star}{\lambda\alpha\rho \Rightarrow_G \lambda\beta\rho}$$

Съобразете, че имаме свойството:

$$\frac{\alpha \Rightarrow_G \beta \qquad \lambda, \rho \in (V \cup \Sigma)^\star}{\lambda\alpha\rho \Rightarrow_G \lambda\beta\rho}$$

Сега е удобно е да дефинираме **извод** на думата β от думата α в граматиката G за ℓ стъпки, което ще означаваме като $\alpha \overset{\ell}{\Rightarrow}_G \beta$, с индукция по броя на стъпките ℓ по следния начин:

В правило (1), нямаме никакви ограничения за λ и ρ. Те са произволни елементи на $(V \cup \Sigma)^\star$, което означава, че може и да са празните думи.

$$\frac{\alpha \in (V \cup \Sigma)^\star}{\alpha \overset{0}{\Rightarrow}_G \alpha} \text{ (рефлексивност)} \qquad \frac{\alpha \Rightarrow_G \beta \qquad \beta \overset{\ell}{\Rightarrow}_G \gamma}{\alpha \overset{\ell+1}{\Rightarrow}_G \gamma} \text{ (транзитивност)}$$

Фигура 3.1: Правила за извод в неограничена граматика

Сега дефинираме релацията $\overset{\star}{\Rightarrow}_G$ като

$$\alpha \overset{\star}{\Rightarrow}_G \beta \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[\, \alpha \overset{\ell}{\Rightarrow}_G \beta \,].$$

Обърнете внимание, че в тази дефиниция на извод не определяме реда, в който прилагаме правилата на граматиката. Също така, понякога ,за удобство, ще пишем просто $\overset{\ell}{\Rightarrow}$ вместо $\overset{\ell}{\Rightarrow}_G$, когато се знае за коя граматика говорим. С други думи, $\overset{\star}{\Rightarrow}_G$ е рефлексивното и транзитивно затваряне на релацията $\overset{1}{\Rightarrow}_G$.

Езикът, който се поражда от граматиката G дефинираме по следния начин:

$$\mathcal{L}(G) \overset{\text{деф}}{=} \{\omega \in \Sigma^\star \mid S \overset{\star}{\Rightarrow}_G \omega\}.$$

За да можем да работим по-удобно с релацията за извод в граматика, ще започнем с няколко основни свойства.

Твърдение 3.1. За произволно естествено число ℓ имаме извода:

$$\frac{\alpha \overset{\ell}{\Rightarrow} \beta \qquad \lambda, \rho \in (V \cup \Sigma)^\star}{\lambda\alpha\rho \overset{\ell}{\Rightarrow} \lambda\beta\rho}$$

Доказателство. Индукция по ℓ.

- $\ell = 0$. Тук всичко е ясно, защото тогава $\alpha = \beta$.
- $\ell > 0$. Според правилата за извод, можем да разбием извода $\alpha \overset{\ell}{\Rightarrow} \beta$ по следния начин:

$$\frac{\alpha \Rightarrow \gamma \qquad \gamma \overset{\ell-1}{\Rightarrow} \beta}{\alpha \overset{\ell}{\Rightarrow} \beta}\ \text{(транзитивност)}$$

Сега като използваме **(И.П.)** получаваме следния извод:

$$\dfrac{\dfrac{\alpha \Rightarrow \gamma}{\lambda\alpha\rho \Rightarrow \lambda\gamma\rho} \quad \dfrac{\gamma \overset{\ell-1}{\Rightarrow} \beta}{\lambda\gamma\rho \overset{\ell-1}{\Rightarrow} \lambda\beta\rho}\ \textbf{(И.П.)}}{\lambda \underbrace{\gamma\alpha'\delta}_{\alpha} \rho \overset{\ell}{\Rightarrow} \lambda\beta\rho}\ \text{(транзитивност)}$$

□

Твърдение 3.2. За произволни ℓ_1 и ℓ_2 имаме извода:

$$\frac{\alpha_1 \overset{\ell_1}{\Rightarrow} \beta_1 \qquad \alpha_2 \overset{\ell_2}{\Rightarrow} \beta_2}{\alpha_1\alpha_2 \overset{\ell_1+\ell_2}{\Rightarrow} \beta_1\beta_2}$$

Доказателство. Индукция по ℓ_1.

- Ако $\ell_1 = 0$, то $\alpha_1 = \beta_1$ и тогава:

$$\text{(\textit{Твърдение 3.1})}\ \frac{\alpha_1 = \beta_1 \qquad \alpha_2 \overset{\ell_2}{\Rightarrow}_G \beta_2}{\alpha_1\alpha_2 \overset{\ell_2}{\Rightarrow}_G \beta_1\beta_2}$$

- Ако $\ell_1 > 0$, то разбиваме извода $\alpha_1 \overset{\ell_1}{\Rightarrow} \beta_1$ по следния начин:

$$\frac{\alpha_1 \Rightarrow \gamma_1 \qquad \gamma_1 \overset{\ell_1-1}{\Rightarrow} \beta_1}{\alpha \overset{\ell_1}{\Rightarrow} \beta_1}\ \text{(транзитивност)}$$

  Сега прилагаме **(И.П.)** и получаваме следния извод:

$$\dfrac{\dfrac{\alpha_1 \Rightarrow \gamma_1}{\alpha_1\alpha_2 \Rightarrow \gamma_1\alpha_2} \qquad \dfrac{\gamma_1 \overset{\ell_1-1}{\Rightarrow} \beta_1 \qquad \alpha_2 \overset{\ell_2}{\Rightarrow} \beta_2}{\gamma_1\alpha_2 \overset{\ell_1-1+\ell_2}{\Rightarrow} \beta_1\beta_2}\ \text{(И.П.)}}{\alpha_1\alpha_2 \overset{\ell_1+\ell_2}{\Rightarrow} \beta_1\beta_2}\ \text{(транзитивност)}$$

□

Твърдение 3.3. За всяко k е изпълнено, че:

$$\frac{\alpha_1 \overset{\ell_1}{\Rightarrow} \beta_1 \qquad \ldots \qquad \alpha_k \overset{\ell_k}{\Rightarrow} \beta_k}{\alpha_1 \cdots \alpha_k \overset{\ell}{\Rightarrow} \beta_1 \cdots \beta_k}\ (\ell = \textstyle\sum_{i=1}^{k} \ell_i)$$

Упътване. Индукция по k. □

Твърдение 3.4. За произволни естествени числа ℓ_1 и ℓ_2 е изпълнено, че:

$$\frac{\alpha \overset{\ell_1}{\Rightarrow} \beta \qquad \beta \overset{\ell_2}{\Rightarrow} \gamma}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow} \gamma}$$

Доказателство. Индукция по ℓ_1.

- $\ell_1 = 0$. Този случай е ясен, защото тогава $\alpha = \beta$ и имаме извода:

$$\text{(рефлексивност)}\ \frac{\alpha = \beta}{\dfrac{\alpha \overset{0}{\Rightarrow} \beta \qquad \beta \overset{\ell_2}{\Rightarrow} \gamma}{\alpha \overset{0+\ell_2}{\Rightarrow} \gamma}}$$

- $\ell_1 > 0$. Да разбием извода $\alpha \overset{\ell_1}{\Rightarrow} \beta$ така:

$$\frac{\alpha \Rightarrow \alpha' \qquad \alpha' \overset{\ell_1-1}{\Rightarrow} \beta}{\alpha \overset{\ell_1}{\Rightarrow} \beta}\ \text{(транзитивност)}$$

Сега можем да приложим индукционното предположение и да получим извода:

$$\dfrac{\alpha \Rightarrow \alpha' \qquad \dfrac{\alpha' \overset{\ell_1-1}{\Rightarrow} \beta \qquad \beta \overset{\ell_2}{\Rightarrow} \gamma}{\alpha' \overset{\ell_1-1+\ell_2}{\Rightarrow} \gamma}\ \textbf{(и.п.)}}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow} \gamma}\ \text{(транзитивност)}$$

□

Следствие 3.1. За произволни естествени числа ℓ_1 и ℓ_2 е изпълнено, че:

$$\dfrac{\alpha \overset{\ell_1}{\Rightarrow} \lambda\beta\rho \qquad \beta \overset{\ell_2}{\Rightarrow} \gamma}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow} \lambda\gamma\rho}$$

3.2 Контекстни граматики

На англ. context-sensitive [HU79, стр. 223]. На български може да се преведат и като контекстнозависими граматики. В йерархията на Чомски това са граматиките от тип 1. В дефиницията, позволяваме и правилото $S \to \varepsilon$, ако искаме да включим ε в езика, като тогава изискваме S да не се среща в дясна страна на правило. По-проста граматика в [Sha08, стр. 202].

Разглеждането на контекстни граматики излиза извън целите на този курс. Добавяме този раздел единствено за пълнота на изложението. Казваме, че $G = (V, \Sigma, R, S)$ е **контекстна граматика**, ако правилата на G са от вида $\lambda A \rho \to \lambda \alpha \rho$, където $\lambda, \rho \in (V \cup \Sigma)^\star$ и $\alpha \in (V \cup \Sigma)^+$.

Пример 3.1. Езикът $L = \{a^n b^n c^n \mid n > 0\}$ е контекстен.

Упътване. Разгледайте контекстната граматика G зададена със следните правила:

$$
\begin{aligned}
&S \to aSBC \mid aBC\\
&CB \to CZ\\
&CZ \to WZ\\
&WZ \to WC\\
&WC \to BC\\
&aB \to ab\\
&bB \to bb\\
&bC \to bc\\
&cC \to cc.
\end{aligned}
$$

Докажете, че за всяко $n > 0$ е изпълено следното:

- $S \overset{n}{\Rightarrow} a^n (BC)^n$;
- $CB \overset{4}{\Rightarrow}_G BC$.
- $(BC)^n \overset{4(n-1)}{\Rightarrow} B^n C^n$;
- $aB^n \overset{n}{\Rightarrow} ab^n$;
- $bC^n \overset{n}{\Rightarrow} bc^n$.

Оттук лесно можем да докажем, че $L \subseteq \mathcal{L}(G)$. □

3.3 Регулярни граматики

Сега ще разгледаме граматики с такъв вид правила, които пораждат точно регулярните (или еквивалентно автоматни) езици. Граматиката $G = \langle V, \Sigma, R, S\rangle$ се нарича **регулярна граматика**, ако всички правила са от вида

$$A \to aB,$$
$$A \to \varepsilon,$$

за произволни $A, B \in V$ и $a \in \Sigma$.

Също така се наричат граматики от тип 3 в йерархията на Чомски [HU79, стр. 217]. Този вид граматики понякога се нарича и дясно-регулярна граматика.

Лема 3.1. За всяка регулярна граматика G съществува недетерминиран краен автомат $\mathcal{N}$, такъв че $\mathcal{L}(G) = \mathcal{L}(\mathcal{N})$.

Упътване. Нека $G = \langle V, \Sigma, R, S\rangle$ и $V = \{A_0, \ldots, A_k\}$, където $S = A_0$. Тогава дефинираме $\mathcal{N}$ по следния начин:

- $Q \stackrel{\text{деф}}{=} \{q_0, \ldots, q_k\}$;
- $Q_{\texttt{start}} \stackrel{\text{деф}}{=} \{q_0\}$;
- $F \stackrel{\text{деф}}{=} \{q_i \mid A_i \to \varepsilon\}$;
- $\Delta(q_i, a) \stackrel{\text{деф}}{=} \{q_j \mid A_i \to aA_j \text{ е правило в граматиката}\}$.

Докажете, че $\mathcal{L}(\mathcal{N}) = \mathcal{L}(G)$. □

Лема 3.2. За всеки детерминиран краен автомат $\mathcal{A}$ съществува регулярна граматика G, такава че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(G)$.

Упътване. Нека $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F\rangle$ и $Q = \{q_0, \ldots, q_k\}$, където $q_{\texttt{start}} = q_0$. Тогава дефинираме $G = \langle V, \Sigma, R, S\rangle$ по следния начин:

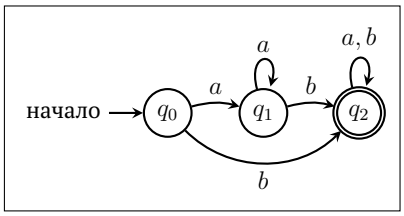
Тази конструкция може да се приложи и за недетерминиран автомат. Единствено трябва да се внимава, ако автоматът има много начални състояния.

- $V \stackrel{\text{деф}}{=} \{A_0, \ldots, A_k\}$;
- $S \stackrel{\text{деф}}{=} A_0$;
- $A_i \to aA_j \stackrel{\text{деф}}{\Leftrightarrow} \delta(q_i, a) = q_j$;
- $A_i \to \varepsilon \stackrel{\text{деф}}{\Leftrightarrow} q_i \in F$.

Докажете, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(G)$. □

Теорема 3.1. Един език е регулярен точно тогава, когато се поражда от регулярна граматика.

Пример 3.2. Да разгледаме отново автомата от *Фигура 2.8*.



Фигура 3.2: $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbf{a}^\star\mathbf{b}(\mathbf{a}+\mathbf{b})^\star)$.

Регулярна граматика G за езика $\mathcal{L}(\mathcal{A})$ можем да дефинираме така:

- $\Sigma = \{a, b\}$;
- На всяко състояние q_i на автомата ще съответства променливата A_i, т.е. $V = \{A_0, A_1, A_2\}$;
- Началната променлива е A_0, защото q_0 е началното състояние на автомата;
- Правилата следват дефиницията на δ функцията:

$$A_0 \to aA_1 \mid bA_2$$
$$A_1 \to aA_1 \mid bA_2$$
$$A_2 \to aA_2 \mid bA_2 \mid \varepsilon.$$

Допълнителни задачи

Задача 3.1. Граматиката $G = (V, \Sigma, R, S)$ се нарича **обобщено дясно-регулярна**, ако всички правила са от вида

$$A \to \omega B,$$
$$A \to \omega$$

за произволни $A, B \in V$ и $\omega \in \Sigma^\star$. Докажете, че един език L е автоматен точно тогава, когато L може да се опише с обобщена дясно-регулярна граматика.

Задача 3.2. Граматиката $G = (V, \Sigma, R, S)$ се нарича **обобщено ляво-регулярна**, ако всички правила са от вида

$$A \to B\omega,$$
$$A \to \omega$$

за произволни $A, B \in V$ и $\omega \in \Sigma^\star$. Докажете, че един език L е автоматен точно тогава, когато L може да се опише с обобщена ляво-регулярна граматика.

3.4 Безконтекстни граматики

В Раздел 3.1 въведохме понятието неограничена граматика. След това видяхме как можем да опишем регулярните езици със специален вид граматики, които нарекохме регулярни граматики. Сега ще разгледаме още един вид граматики, които описват по-широк клас от езици.

Една граматика $G = (V, \Sigma, R, S)$ се нарича **безконтекстна**, ако имаме ограничението, че $R \subseteq V \times (V \cup \Sigma)^\star$. Да повторим дефиницията на релацията $\alpha \Rightarrow_G \beta$ от Раздел 3.1 в частния случай, когато граматиката е безконтекстна.

В [PL98] дефиницията е различна. Там $\Sigma \subseteq V$. На англ. *context-free grammar*. Други срещани наименования на български са *контекстносвободна*, *контекстнонезависима*. Тук всички правила са от вида $A \to \alpha$, където $\alpha \in (V \cup \Sigma)^\star$. В частност имаме, че:

$$\frac{(A, \alpha) \in R}{A \Rightarrow_G \alpha}$$

$$\frac{(A, \alpha) \in R \qquad \lambda, \rho \in (V \cup \Sigma)^\star}{\lambda A \rho \Rightarrow_G \lambda \alpha \rho}$$

Нека официално да обявим, че един език L се нарича **безконтекстен**, ако съществува безконтекстна граматика G, за която $L = \mathcal{L}(G) = \{\omega \in \Sigma^\star \mid S \overset{\star}{\Rightarrow} \omega\}$.

Като частен случай на *Твърдение 3.4* получаваме следното свойство.

Твърдение 3.5. За произволни естествени числа ℓ_1 и ℓ_2 е изпълнено, че:

$$\frac{\alpha \overset{\ell_1}{\Rightarrow} \rho B \delta \qquad B \overset{\ell_2}{\Rightarrow} \beta}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow} \rho \beta \delta,}$$

Пример 3.3. Да разгледаме безконтекстната граматика G, която има следните правила:

$$S \to_G AS \mid \varepsilon$$
$$A \to_G aAb \mid ab.$$

Да видим защо думата $aabbab \in \mathcal{L}(G)$. Ако следваме формално правилата за извод, получаваме следното:

За момента не е ясно как можем да проверим, че примерно думата $abbba \notin \mathcal{L}(G)$. Този въпрос ще разгледаме по-нататък.

$$\dfrac{\dfrac{\dfrac{(S, AS) \in R}{S \Rightarrow_G AS} \qquad \dfrac{\dfrac{}{A \overset{0}{\Rightarrow}_G A} \qquad \dfrac{\dfrac{\dfrac{(S, \varepsilon) \in R}{S \Rightarrow_G \varepsilon} \qquad \dfrac{}{\varepsilon \overset{0}{\Rightarrow}_G \varepsilon}}{S \overset{1}{\Rightarrow}_G \varepsilon} \qquad \dfrac{\dfrac{(S, AS) \in R}{S \Rightarrow_G AS} \qquad \dfrac{}{AS \overset{0}{\Rightarrow}_G AS}}{S \overset{1}{\Rightarrow}_G AS}\ (\textit{Твърдение 3.5})}{S \overset{2}{\Rightarrow}_G A}\ (\textit{Твърдение 3.2})}{AS \overset{2}{\Rightarrow}_G AA}\ (\text{транзитивност})}{S \overset{3}{\Rightarrow}_G AA}$$

Освен това имаме и следния формален извод:

$$\dfrac{\dfrac{(A, aAb) \in R}{A \Rightarrow_G aAb} \qquad \dfrac{(A, ab) \in R}{aAb \overset{1}{\Rightarrow}_G aabb}}{A \overset{2}{\Rightarrow}_G aabb}$$

Аналогично,

$$\dfrac{\dfrac{(A, ab) \in R}{A \Rightarrow_G ab} \qquad \dfrac{}{ab \overset{0}{\Rightarrow}_G ab}}{A \overset{1}{\Rightarrow}_G ab}$$

Обединявайки всичко, получаваме:

$$\dfrac{\dfrac{S \overset{3}{\Rightarrow}_G AA \qquad A \overset{2}{\Rightarrow}_G aabb}{S \overset{5}{\Rightarrow}_G aabbA} \text{ (Твърдение 3.5)} \qquad A \overset{1}{\Rightarrow}_G ab}{S \overset{6}{\Rightarrow}_G aabbab} \text{ (Твърдение 3.5)}$$

Можем да приложим правилата за извод в различен ред и пак да получим същия краен резултат. Например:

$$\dfrac{\dfrac{\dfrac{(S, AS) \in R}{S \Rightarrow_G AS} \qquad \dfrac{A \overset{2}{\Rightarrow} aabb}{AS \overset{2}{\Rightarrow} aabbS}}{S \overset{3}{\Rightarrow}_G aabbS} \text{ (Твърдение 3.1)} \qquad S \overset{2}{\Rightarrow}_G A}{S \overset{5}{\Rightarrow}_G aabbA} \text{ (Твърдение 3.5)} \qquad A \overset{1}{\Rightarrow}_G ab}{S \overset{6}{\Rightarrow}_G aabbab} \text{ (Твърдение 3.5)}$$

Следващото твърдение ни дава едно свойство, което е вярно за безконтекстни граматики, но не и за неограничени граматики.

Тук $\gamma_1, \gamma_2, \beta \in (V \cup \Sigma)^\star$.

Твърдение 3.6. Нека G е безконтекстна граматика и нека γ_1 и γ_2 са непразни думи, за които $\gamma_1\gamma_2 \overset{\ell}{\Rightarrow} \beta$. Тогава съществуват числа ℓ_1, ℓ_2 и думи β_1 и β_2, за които: $\gamma_1 \overset{\ell_1}{\Rightarrow} \beta_1$ и $\gamma_2 \overset{\ell_2}{\Rightarrow} \beta_2$ и $\beta = \beta_1\beta_2$ и $\ell = \ell_1 + \ell_2$.

Доказателство. Индукция по дължината на извода ℓ.

- Нека $\ell = 0$. Тогава $\beta_1 = \gamma_1$ и $\beta_2 = \gamma_2$.
- Нека $\ell > 0$. Тогава да разгледаме следната ситуация:

$$\dfrac{\dfrac{(A, \alpha) \in R}{\lambda A \rho \Rightarrow_G \lambda\alpha\rho} \qquad \lambda\alpha\rho \overset{\ell-1}{\Rightarrow} \beta}{\underbrace{\lambda A \rho}_{\gamma_1\gamma_2} \overset{\ell}{\Rightarrow} \beta}$$

Случаят, когато A е част от γ_2 е аналогичен.

Без ограничение на общността, нека променливата A е част от γ_1. Това означава, че $\gamma_1 = \lambda A \rho_1$ и $\rho = \rho_1\gamma_2$. Сега можем да приложим индукционното предположение и да заключим, че съществува представяне на β като $\beta = \beta_1\beta_2$ със следния извод:

$$\dfrac{\dfrac{(A,\alpha)\in R}{\lambda A\rho_1 \Rightarrow_G \lambda\alpha\rho_1} \quad \dfrac{\overbrace{\lambda\alpha\rho_1}^{\gamma_1'}\gamma_2 \overset{\ell-1}{\Rightarrow} \beta}{\lambda\alpha\rho_1 \overset{\ell_1'}{\Rightarrow} \beta_1 \,\&\, \gamma_2 \overset{\ell_2}{\Rightarrow} \beta_2}\ \textbf{(и.п.)}}{\underbrace{\lambda A\rho_1}_{\gamma_1} \overset{\ell_1'+1}{\Rightarrow} \beta_1 \,\&\, \gamma_2 \overset{\ell_2}{\Rightarrow} \beta_2}$$

□

Твърдение 3.7. Нека G е безконтекстна граматика и нека $X_1 \cdots X_k \overset{\ell}{\Rightarrow}_G \beta$, където $X_i \in V \cup \Sigma$ и $k \geq 2$. Тогава съществуват думи $\beta_1, \ldots, \beta_k$, такива че за $i = 1, \ldots, k$ е изпълнено, че $X_i \overset{\ell_i}{\Rightarrow} \beta_i$, където $\beta = \beta_1 \cdots \beta_k$ и $\ell = \sum_{i=1}^{k} \ell_i$.

Упътване. Пълна индукция по k като използвате *Твърдение 3.6*. □

Тук е възможно $X_i = a \in \Sigma$. Тогава $a \overset{0}{\Rightarrow} a$ и $\beta_i = a$.

Следващото твърдение е важно, но е трудно да не пропуснем доказателството му.

Твърдение 3.8. Безконтекстните езици са затворени относно операциите обединение, конкатенация и звезда на Клини.

Оттук можем да извлечем второ доказателство на твърдението, че всеки регулярен език е безконтекстен.

Твърдение 3.9. Всеки регулярен език е безконтекстен.

Упътване. Можем да подходим поне по два начина.

- Да проведем индукция по построението на регулярните езици и да докажем така, че всеки регулярен език е безконтекстен.
- Просто да съобразим, че всяка регулярна граматика е всъщност и безконтекстна.

□

3.5 Синтактични дървета

На английски се нарича parse tree, syntax tree, derivation tree. Обикновено, например в [PL98, стр. 123], дават рекурсивна дефиниция на синтактично дърво. A parse tree is a graphical representation of a derivation thet filters out the order in which productions are applied to replace nonterminals. [ALSU07]

- Нека фиксираме граматиката $G = (\Sigma, V, S, R)$ и

$$b \stackrel{\text{деф}}{=} \max\{|\gamma| \mid A \to_G \gamma \text{ е правило в } G\}.$$

- Двойката $P = (T, \lambda)$ се нарича **дърво на извод** съвместимо с G, ако са изпълнени свойствата:
  - T е непразно крайно *оптимално* дърво и $T \subseteq \{0, 1, \ldots, b-1\}^\star$.
  - Фукцията $\lambda : T \to V \cup \Sigma \cup \{\varepsilon\}$ асоциира етикет с всеки елемент на дървото. Нека означим тези етикети с $X_\alpha \stackrel{\text{деф}}{=} \lambda(\alpha)$.
  - Коренът на дървото винаги има етикет елемент на V или Σ, т.е. $\lambda(\varepsilon) \in V \cup \Sigma$.
  - Ако $\alpha \in T$ и $|\mathtt{succ}_T(\alpha)| = k + 1$, за някое $k \in \mathbb{N}$, то има правило в граматиката

С други думи, $\lambda(\alpha) \to_G \lambda(\alpha 0) \cdots \lambda(\alpha k)$.

$$X_\alpha \to_G X_{\alpha 0} X_{\alpha 1} \cdots X_{\alpha k},$$

Това свойство е ключово, защото то ни казва, че дървото е съвместимо с правилата на граматиката G.

    където $X_{\alpha i} \in V \cup \Sigma$.
  - Имаме частен случай за $\alpha \in T$ и $|\mathtt{succ}_T(\alpha)| = 1$. Тогава позволяваме да имаме $X_{\alpha 0} = \varepsilon$, ако имаме правилото в граматиката

$$X_\alpha \to_G \varepsilon.$$

- За дървото на извод P, нека $\mathtt{root}(P) \stackrel{\text{деф}}{=} X_\varepsilon$.

- Нека $\alpha_0, \alpha_1, \ldots, \alpha_k$ са всички думи от множеството $\mathtt{leaves}(T)$ подредени във възходящ ред относно лексикографската наредба. Тогава

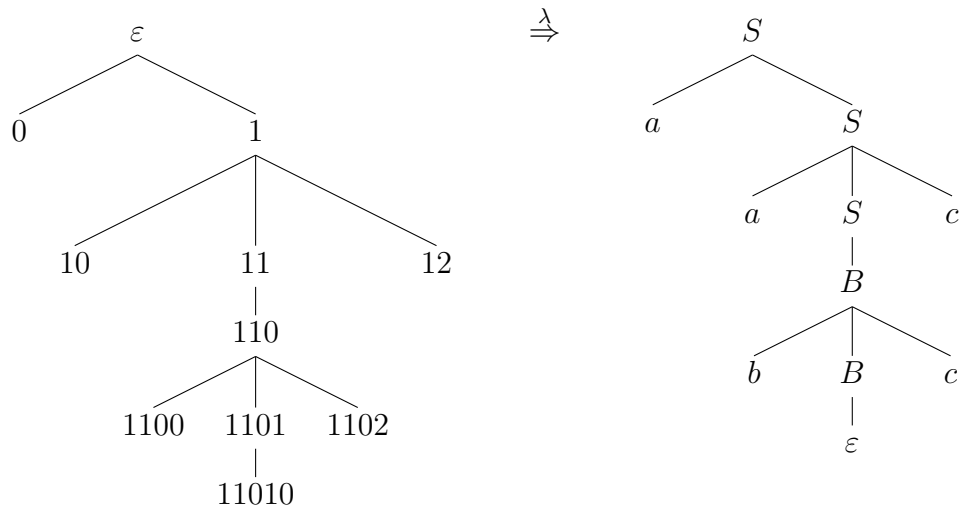
$$\mathtt{yield}(P) \stackrel{\text{деф}}{=} X_{\alpha_0} X_{\alpha_1} \cdots X_{\alpha_k}.$$

Пример 3.4. Да разгледаме граматиката G зададена със следните правила:

$$S \to aS \mid aSc \mid B$$
$$B \to bB \mid bBc \mid \varepsilon.$$

По-нататък ще докажем, че езикът на граматиката G е $\{a^n b^k c^\ell \mid n + k \geq \ell\}$.

Да разгледаме дървото на извод $P = (T, \lambda)$, където:



Фигура 3.3: Дърво на извод за думата $aabcc$.

Имаме, че:

- $\texttt{height}(P) = 5$;
- $\texttt{leaves}(P) = \{0, 10, 1100, 11010, 1102, 12\}$;
- $\texttt{yield}(P) = aab\varepsilon cc = aabcc$.
- $\texttt{succ}_T(110) = \{1100, 1101, 1102\}$;
- $\lambda(\varepsilon) = S$;
- $\lambda(0) = a$, $\lambda(1) = S$;
- Ясно е, че $\underbrace{\lambda(\varepsilon)}_{S} \to_G \underbrace{\lambda(0)\lambda(1)}_{aS}$;
- $\lambda(10) = a$, $\lambda(11) = S$, $\lambda(12) = c$;
- Ясно е, че $\underbrace{\lambda(1)}_{S} \to_G \underbrace{\lambda(10)\lambda(11)\lambda(12)}_{aSc}$;
- $\lambda(110) = B$;
- Ясно е, че $\underbrace{\lambda(11)}_{S} \to_G \underbrace{\lambda(110)}_{B}$;
- $\lambda(1100) = b$, $\lambda(1101) = B$, $\lambda(1102) = c$;
- Ясно е, че

$$\underbrace{\lambda(110)}_{B} \to_G \underbrace{\lambda(1100)\lambda(1101)\lambda(1102)}_{bBc};$$

- $\lambda(11010) = \varepsilon$;
- Ясно е, че $\underbrace{\lambda(1101)}_{B} \to_G \underbrace{\lambda(11010)}_{\varepsilon}$;

От всичко по-горе следва, че $P = (T, \lambda)$ е дърво на извод за думата $aabcc$ в граматиката G.

3.6 Извод върху синтактично дърво

Не знам да има учебник, в който формално да е въведена тази релация. Тя е удобна най-вече за решаване на задачи, както и за доказателството на лемата за покачването.

Определение 3.1. За произволна безконтекстна граматика G, дефинираме релацията $X \overset{\ell}{\lhd} \alpha$, където $X \in V \cup \Sigma$ и $\alpha \in (V \cup \Sigma)^\star$, по следния начин:

$$\frac{}{X \overset{0}{\lhd} X} \text{ правило (0)}$$

$$(\ell = \sup\{\ell_1, \ldots, \ell_n\}) \quad \frac{X \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \gamma_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \gamma_n}{X \overset{\ell+1}{\lhd} \gamma_1 \cdots \gamma_n} \text{ правило (1)}$$

Да напомним, че по дефиниция, ако $n = 0$, то $X_1 \cdots X_n = \varepsilon$. Освен това, понеже $\sup(A)$ означава най-малкото естествено число по-голямо от всички елементи на A, то $\sup(\emptyset) = 0$. Това означава, че ако в граматиката имаме правилото $A \to_G \varepsilon$, то според правило (1) получаваме, че $A \overset{1}{\lhd} \varepsilon$.

Съобразете, че имаме:

$$\frac{X \to_G \gamma}{X \overset{1}{\lhd} \gamma.}$$

Да дефинираме $\overset{\star}{\lhd}$ по следния начин:

$$X \overset{\star}{\lhd} \gamma \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[X \overset{\ell}{\lhd} \gamma].$$

Обърнете внимание също, че тази релация е рефликсивна, но не е транзитивна!

Релацията $\overset{\star}{\lhd}$ е удобна, когато не се интересуваме от реда, по който прилагаме правилата за извод в една безконтекстна граматика.

Лема 3.3. Нека G е безконтекстна граматика, $X \in V \cup \Sigma$ и $\beta \in (V \cup \Sigma)^\star$. Тогава ако $X \overset{\star}{\Rightarrow} \beta$, то $X \overset{\star}{\lhd} \beta$.

Доказателство. С пълна индукция по ℓ ще докажем, че ако $X \overset{\ell}{\Rightarrow} \beta$, то $X \overset{\star}{\lhd} \beta$.

- $\ell = 0$, т.е. $X \overset{0}{\Rightarrow} X$. Тогава е ясно, че $X \overset{\star}{\lhd} X$.
- Нека $\ell > 0$ и $X \overset{\ell}{\Rightarrow} \beta$. Според правилата на извод в граматика имаме извода

$$\frac{X \to_G X_1 \cdots X_k \qquad X_1 \cdots X_k \overset{\ell-1}{\Rightarrow} \beta}{X \overset{\ell}{\Rightarrow} \beta}$$

Естествено, че е възможно някои X_i да са букви от Σ. Тогава $\beta_i = X_i$ и $X_i \overset{0}{\Rightarrow}_G \beta_i$.

Щом имаме, че $X_1 \cdots X_k \overset{\ell-1}{\Rightarrow} \beta$, от *Твърдение 3.7* знаем, че съществува разбиване на β на $k+1$ части, така че:

– $\beta = \beta_1 \cdots \beta_k$;

- $X_i \overset{\ell_i}{\Rightarrow} \beta_i$, за всяко $i = 1, \ldots, k$;
- $\ell - 1 = \sum_{i=1}^{k} \ell_i$.

Понеже $\ell_i < \ell$ за всяко $i = 1, \ldots, k$, получаваме следното:

$$\dfrac{X \to_G X_1 \cdots X_k \qquad \dfrac{X_1 \overset{\ell_1}{\Rightarrow} \beta_1}{X_1 \overset{\star}{\lhd} \beta_1}\ \text{(И.П.)} \qquad \cdots \qquad \dfrac{X_k \overset{\ell_k}{\Rightarrow} \beta_k}{X_k \overset{\star}{\lhd} \beta_k}\ \text{(И.П.)}}{X \overset{\star}{\lhd} \underbrace{\beta_1 \cdots \beta_k}_{\beta}}\ \text{правило (1)}$$

□

Лема 3.4. Нека G е безконтекстна граматика, $X \in V \cup \Sigma$ и $\gamma \in (V \cup \Sigma)^\star$. Тогава ако $X \overset{\star}{\lhd} \gamma$, то $X \overset{\star}{\Rightarrow} \gamma$.

Доказателство. С пълна индукция по ℓ ще докажем, че ако $X \overset{\ell}{\lhd} \gamma$, то $X \overset{\star}{\Rightarrow} \gamma$.

- Нека $\ell = 0$. Това означава, че $X \overset{0}{\lhd} X$. Ясно е, че $X \overset{\star}{\Rightarrow} X$.
- Нека $\ell > 0$. Тогава имаме следното:

$$\dfrac{X \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \gamma_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \gamma_n}{X \overset{\ell}{\lhd} \underbrace{\gamma_1 \cdots \gamma_n}_{\gamma}}\ (\ell = 1 + \sup\{\ell_1, \ldots, \ell_n\})$$

Понеже $\ell_i < \ell$ за всяко $i = 1, \ldots, n$, получаваме следното:

$$\dfrac{X \to_G X_1 \cdots X_n \qquad \dfrac{\dfrac{X_1 \overset{\ell_1}{\lhd} \gamma_1}{X_1 \overset{\star}{\Rightarrow} \gamma_1}\ \text{(И.П.)} \qquad \cdots \qquad \dfrac{X_n \overset{\ell_n}{\lhd} \gamma_n}{X_n \overset{\star}{\Rightarrow} \gamma_n}\ \text{(И.П.)}}{X_1 \cdots X_n \overset{\star}{\Rightarrow} \gamma_1 \cdots \gamma_n}\ \text{(Твърдение 3.3)}}{X \overset{\star}{\Rightarrow} \underbrace{\gamma_1 \cdots \gamma_n}_{\gamma}}$$

□

Комбинирайки предишните две леми получаваме следната теорема.

Теорема 3.2. Нека G е безконтекстна граматика, $X \in V \cup \Sigma$ и $\gamma \in (V \cup \Sigma)^\star$. Тогава $X \overset{\star}{\lhd} \gamma$ точно тогава, когато $X \overset{\star}{\Rightarrow} \gamma$. В частност,

$$\mathcal{L}(G) = \{\alpha \in \Sigma^\star \mid S \overset{\star}{\lhd} \alpha\}.$$

Следващото твърдение ни казва, че има пряка връзка между релацията $\overset{\star}{\lhd}$ и синтактичните дървета.

Твърдение 3.10. Нека G е безконтекстна граматика. Тогава $X \overset{\ell}{\lhd} \gamma$ точно тогава, когато съществува дърво на извод P съвместимо с G, за което $\texttt{root}(P) = X$, $\texttt{yield}(P) = \gamma$ и $\texttt{height}(P) = \ell$.

Упътване. Индукция по ℓ. □

Еднозначни граматики

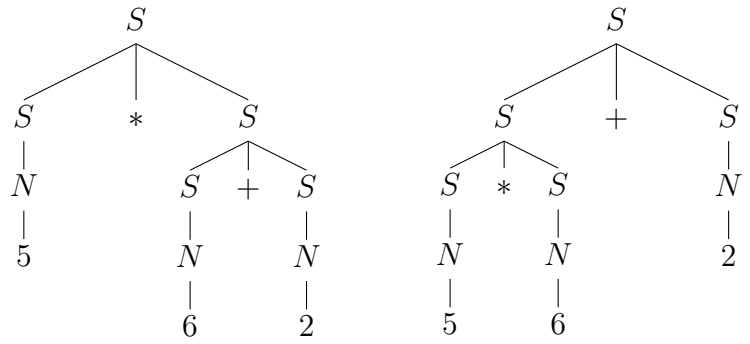
Добре е да обърнем внимание, че е възможно, ако $A \overset{\ell}{\lhd} \alpha$, то да има няколко синтактични дървета с корен променливата A, които да извеждат думата α и да имат една и съща височина ℓ.

Пример 3.5. Да разгледаме граматика G_0 с правила

$$S \to S + S \mid S * S \mid (S)$$
$$N \to 0 \mid 1 \mid \ldots \mid 9$$

Такъв вид граматики се наричат *нееднозначни* (на англ. *ambiguous*). Ако всяка дума от езика има единствено синтактично дърво, то тази граматика се нарича *еднозначна* (на англ. *unambiguous*). От практическа гледна точка е важно свойство дали една граматика е еднозначна или не.

В тази граматика нямаме приоритет между $+$ или $*$. Например, думата $5 * 6 + 2$ има две различни синтактични дървета P_1 и P_2, като и двете имат височина 4 и $\texttt{yield}(P_1) = \texttt{yield}(P_2) = 5 * 6 + 2$.



Фигура 3.4: Две синтактични дървета в граматиката G_0 за думата $5 * 6 + 2$.

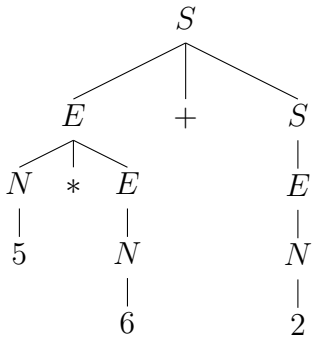
Искаме да модифицираме G_0, така че $*$ да има по-висок приоритет спрямо $+$. За целта ще разгледаме граматиката G_1, която ще поражда същия език като G_0, със следните правила:

✍ Колко различни синтактични дървета за думата $1 + 5 * 6 + 2$ може да намерите?

$$
\begin{aligned}
S &\to E + S \mid E \\
E &\to N * E \mid N \mid (S) * E \mid (S) \\
N &\to 0 \mid 1 \mid \ldots \mid 9
\end{aligned}
$$

Сега думата $5 * 6 + 2$ има само едно дърво на извод. Тук операцията $*$ е с по-висок приоритет в сравнение с операцията $+$.

$*$ има по-висок приоритет от $+$, защото $*$ се среща по-близо до листата.



Фигура 3.5: Единствено синтактично дърво в граматиката G_1 за думата $5 * 6 + 2$.

Ще завършим този раздел като формулираме един важен резултат, за който все още нямаме сили да докажем.

Доказателството е в *Следствие 4.10*.

Теорема 3.3. Не съществува алгоритъм, който да разпознава дали дадена безконтекстна граматика е еднозначна или не.

Примерни задачи

Да напомним, че в Раздел 2.10 дефинирахме език $\mathcal{L}_{\mathcal{A}}(q)$. Сега ще дефинираме език $\mathcal{L}_G(A)$, за произволна променлива A по следния начин:

$$\mathcal{L}_G(A) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid A \overset{\star}{\lhd} \alpha\}.$$

Ясно е, че $\mathcal{L}(G) = \mathcal{L}_G(S)$. Нека да дефинираме и *апроксимациите* $\mathcal{L}^\ell_G(A)$ на езика $\mathcal{L}_G(A)$ по следния начин:

$$\mathcal{L}^\ell_G(A) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid A \overset{\leq \ell}{\lhd} \alpha\}.$$

Задача 3.3. Нека G е безконтекстна граматика и A е променлива в G. Докажете, че следните свойства са изпълнени:

- $\mathcal{L}^0_G(A) = \emptyset$;
- $\mathcal{L}^\ell_G(A)$ е краен език за всяко ℓ;
- $\mathcal{L}^\ell_G(A) \subseteq \mathcal{L}^{\ell+1}_G(A)$ за всяко ℓ;
- $\mathcal{L}_G(A) = \bigcup_{\ell \geq 0} \mathcal{L}^\ell_G(A)$.

Следната характеризация на езиците $\mathcal{L}^\ell_G(A)$ ще е удобна за нас, когато искаме да докажем, че една безконтекстна граматика разпознава даден език.

Нека да напомним, че

$\bigcup\{\{1,2\},\{3\}\} = \{1,2\} \cup \{3\} = \{1,2,3\}$.

Твърдение 3.11. Нека G е безконтекстна граматика и A е променлива в G. Тогава имаме следното:

$\mathcal{L}^0_G(A) = \emptyset$

$\mathcal{L}^{\ell+1}_G(A) = \bigcup\{\{\alpha_0\} \cdot \mathcal{L}^\ell_G(A_1) \cdots \{\alpha_{n-1}\} \cdot \mathcal{L}^\ell_G(A_n) \cdot \{\alpha_n\} \mid \quad A \to_G \alpha_0 A_1 \cdots \alpha_{n-1} A_n \alpha_n$ е правило в G, където $\alpha_i \in \Sigma^\star$ за $i \leq n\}$.

Доказателство. Доказателството протича с индукция по ℓ. Твърдението очевидно е изпълнено за $\ell = 0$. За индукционната стъпка, първо ще докажем включването $\subseteq$.Нека сега да разгледаме дума $\alpha \in \mathcal{L}^{\ell+1}_G(A)$, т.е. $A \overset{\leq \ell+1}{\lhd} \alpha$. Тогава имаме следния извод:

$$\frac{A \to_G \alpha_0 B_1 \cdots \alpha_{n-1} B_n \alpha_n \qquad B_1 \overset{\leq \ell}{\lhd} \beta_1 \qquad \cdots \qquad B_n \overset{\leq \ell}{\lhd} \beta_n}{A \overset{\leq \ell+1}{\lhd} \underbrace{\alpha_0 \beta_1 \cdots \alpha_{n-1} \beta_n \alpha_n}_{\alpha}}$$

Понеже $B_i \overset{\leq \ell}{\lhd} \beta_i$, то от **(И.П.)** имаме, че $\beta_i \in \mathcal{L}^\ell_G(B_i)$ за всяко $i = 1, 2, \ldots, n$. Тогава

$$\alpha \in \bigcup\{\{\alpha_1\} \cdot \mathcal{L}^\ell_G(A_1) \cdots \{\alpha_n\} \cdot \mathcal{L}^\ell_G(A_n) \cdot \{\alpha_{n+1}\} \mid A \to_G \alpha_1 A_1 \cdots \alpha_n A_n \alpha_{n+1}\}$$

Сега преминаваме към доказателството на включването $\supseteq$. Нека вземем дума α принадлежаща на дясното множество. Това означава, че можем да представим α като $\alpha = \alpha_0 \beta_1 \cdots \alpha_{n-1} \beta_n \alpha_n$, където $A \to_G \alpha_0 B_1 \cdots \alpha_{n-1} B_n \alpha_n$, $\beta_i \in \mathcal{L}^\ell_G(B_i)$ и $\alpha = \alpha_0 \beta_1 \cdots \alpha_{n-1} \beta_n \alpha_n$. Получаваме следния извод:

$$\frac{A \to_G \alpha_0 B_1 \cdots \alpha_{n-1} B_n \alpha_n \qquad \dfrac{\beta_1 \in \mathcal{L}^\ell_G(B_1)}{B_1 \overset{\leq \ell}{\lhd} \beta_1}\ \textbf{(И.П.)} \qquad \cdots \qquad \dfrac{\beta_n \in \mathcal{L}^\ell_G(B_n)}{B_n \overset{\leq \ell}{\lhd} \beta_n}\ \textbf{(И.П.)}}{A \overset{\leq \ell+1}{\lhd} \underbrace{\alpha_0 \beta_1 \cdots \alpha_{n-1} \beta_n \alpha_n}_{\alpha}}$$

□

Следствие 3.2. Нека G е безконтекстна граматика и A е променлива в G. Тогава

$$\mathcal{L}_G(A) = \bigcup\{\{\alpha_1\} \cdot \mathcal{L}_G(A_1) \cdots \{\alpha_n\} \cdot \mathcal{L}_G(A_n) \cdot \{\alpha_{n+1}\} \mid A \to_G \alpha_1 A_1 \cdots \alpha_n A_n \alpha_{n+1}\}.$$

Упътване. Използвайте, че $\mathcal{L}_G(A) = \bigcup_\ell \mathcal{L}_G^\ell(A)$. □

Като първи пример, нека да видим, че съществува безконтекстен език, който не е регулярен.

Да напомним, че вече видяхме, че езикът $L = \{a^n b^n \mid n \in \mathbb{N}\}$ не е регулярен.

Пример 3.6. Да разгледаме безконтекстната граматика G зададена със следните правила:

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

Да разгледаме първите няколко члена на редицата от езици $\mathcal{L}_G^\ell(S)$:

$$\begin{aligned}
&\mathcal{L}_G^0(S) = \emptyset \\
&\mathcal{L}_G^1(S) = \{a\} \cdot \emptyset \cdot \{b\} \cup \{\varepsilon\} = \{\varepsilon\} \\
&\mathcal{L}_G^2(S) = \{a\} \cdot \{\varepsilon\} \cdot \{b\} \cup \{\varepsilon\} = \{\varepsilon, ab\} \\
&\mathcal{L}_G^3(S) = \{a\} \cdot \{\varepsilon, ab\} \cdot \{b\} \cup \{\varepsilon\} = \{\varepsilon, ab, aabb\} = \{a^n b^n \mid n < 3\} \\
&\vdots
\end{aligned}$$

Лесно се съобразява, че $\mathcal{L}_G^\ell(S) = \{a^n b^n \mid n < \ell\}$. Заключаваме, че:

$$\mathcal{L}(G) = \mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) = \{a^n b^n \mid n \in \mathbb{N}\}.$$

Пример 3.7. Да разгледаме безконтекстната граматика G зададена със следните правила:

$$\begin{aligned}
S &\to aSc \mid B \\
B &\to bBc \mid \varepsilon.
\end{aligned}$$

Да видим защо $\mathcal{L}(G) = \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$. Първо ще докажем *коректност* на граматиката. Това означава, че G не генерира думи извън езика, т.е. $\mathcal{L}_G(S) \subseteq \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$. За да направим това обаче, трябва да знаем също така и $\mathcal{L}_G(B)$. Формално казано, трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}_G^\ell(S) \subseteq \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\} \tag{3.1}$$
$$\mathcal{L}_G^\ell(B) \subseteq \{b^k c^k \mid k \in \mathbb{N}\}. \tag{3.2}$$

Това ще направим с индукция по ℓ. Да напомним, че според *Твърдение 3.11* имаме следните връзки:

$$\begin{aligned}
&\mathcal{L}_G^0(S) = \emptyset \\
&\mathcal{L}_G^{\ell+1}(S) = \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\} \cup \mathcal{L}_G^\ell(B) \\
&\mathcal{L}_G^0(B) = \emptyset \\
&\mathcal{L}_G^{\ell+1}(B) = \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\} \cup \{\varepsilon\}.
\end{aligned}$$

Очевидно е, че *Свойство (3.1)* и *Свойство (3.2)* са изпълнени за $\ell = 0$. Да примем, че имаме *Свойство (3.1)* и *Свойство (3.2)* за някое ℓ. Ще докажем свойствата и за $\ell + 1$.

Обърнете внимание, че не можем да докажем *Свойство (3.1)* независимо от *Свойство (3.2)*.

- Първо, нека $\alpha \in \mathcal{L}_G^{\ell+1}(S)$. Имаме два случая.
  - Нека $\alpha \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\}$. От **(И.П.)** следва, че $\alpha = a^{n+1} b^k c^{n+k+1}$ за някои естествени числа n и k. Тогава е ясно, че $\alpha \in \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$.

- Нека $\alpha \in \mathcal{L}_G^\ell(B)$. От **(И.П.)** следва, че $\alpha \in \{b^k c^k \mid k \in \mathbb{N}\} \subseteq \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$.

- Второ, нека $\alpha \in \mathcal{L}_G^{\ell+1}(B)$. Имаме два случая.
  - Нека $\alpha \in \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\}$. От **(И.П.)** следва, че $\alpha = b^{k+1}c^{k+1}$ за някое естествено число k. Тогава е ясно, че $\alpha \in \{b^k c^k \mid k \in \mathbb{N}\}$.
  - Нека $\alpha = \varepsilon$. В този случай също е ясно, че $\alpha \in \{b^k c^k \mid k \in \mathbb{N}\}$.

Оттук заключаваме, че $\mathcal{L}_G(S) \subseteq \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$.

Сега да разгледаме пълнота на граматиката, което означава, че всяка дума от езика се генерира от G. С други думи, $\{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\} \subseteq \mathcal{L}_G(S)$. Първо ще докажем, че

$$\{b^k c^k \mid k \in \mathbb{N}\} \subseteq \mathcal{L}_G(B). \tag{3.3}$$

Това ще направим с *пълна* индукция по дължината на думата. Да разгледаме произволна дума $\alpha \in \{b^k c^k \mid k \in \mathbb{N}\}$.

- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Ясно е, че $\alpha \in \mathcal{L}_G(B)$.
- Нека $|\alpha| > 0$, т.е. $\alpha = b^{k+1}c^{k+1}$. От **(И.П.)** за *Свойство (3.3)* имаме $\alpha \in \{b\} \cdot \mathcal{L}_G(B) \cdot \{c\} \subseteq \mathcal{L}_G(B)$.

Вече сме готови да докажем, че:

$$\{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\} \subseteq \mathcal{L}_G(S). \tag{3.4}$$

Това включване пак ще докажем с *пълна* индукция по дължината на думата. Да разгледаме произволна дума $\alpha \in L$.

- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Ясно е, че $\alpha \in \mathcal{L}_G(B) \subseteq \mathcal{L}_G(S)$.
- Нека сега $|\alpha| > 0$, т.е. $\alpha = a^n b^k c^{n+k}$, където $n > 0$ или $k > 0$. Да разгледаме два случая.
  - Нека $n = 0$. Тогава $\alpha = b^k c^k$ и $k > 0$. Тогава от *Свойство (3.3)* следва, че $\alpha \in \mathcal{L}_G(B) \subseteq \mathcal{L}_G(S)$.
  - Нека $n > 0$. Тогава от **(И.П.)** за *Свойство (3.4)* имаме $\alpha \in \{a\} \cdot \mathcal{L}_G(S) \cdot \{c\} \subseteq \mathcal{L}_G(S)$.

Доказахме коректност и пълнота на граматиката и следователно $\mathcal{L}(G) = \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$.

Пример 3.8. Нека да видим защо езикът $L = \{a^m b^n c^k \mid m + n \geq k\}$ е безконтекстен. Да разгледаме граматиката G с правила

$$S \to aSc \mid aS \mid B$$
$$B \to bBc \mid bB \mid \varepsilon.$$

От *Твърдение 3.11* имаме, че:

$$\begin{aligned}
\mathcal{L}_G^0(S) &= \emptyset \\
\mathcal{L}_G^{\ell+1}(S) &= \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\} \cup \{a\} \cdot \mathcal{L}_G^\ell(S) \cup \mathcal{L}_G^\ell(B) \\
\mathcal{L}_G^0(B) &= \emptyset \\
\mathcal{L}_G^{\ell+1}(B) &= \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\} \cup \{b\} \cdot \mathcal{L}_G^\ell(B) \cup \{\varepsilon\}.
\end{aligned}$$

Тези две свойства ще бъдат нашето **(И.П.)** . Очевидно е, че те са изпълнени за $\ell = 0$.

Да предположим, че за произволно естествено число ℓ е изпълнено следното:

$$\mathcal{L}_G^\ell(S) \subseteq \{a^n b^m c^k \mid n + m \geq k\} \tag{3.5}$$
$$\mathcal{L}_G^\ell(B) \subseteq \{b^m c^k \mid m \geq k\}. \tag{3.6}$$

Ще докажем, че

$$\mathcal{L}_G^{\ell+1}(S) \subseteq \{a^n b^m c^k \mid n + m \geq k\}$$
$$\mathcal{L}_G^{\ell+1}(B) \subseteq \{b^m c^k \mid m \geq k\}.$$

За първото включване, да разгледаме произволна дума $\alpha \in \mathcal{L}_G^{\ell+1}(S)$. Имаме три случая:

- Ако $\alpha \in \mathcal{L}_G^\ell(B)$, то от **(И.П.)** имаме, че
$$\alpha \in \{b^m c^k \mid m \geq k\} \subseteq \{a^n b^m c^k \mid n + m \geq k\}.$$
- Ако $\alpha \in \{a\} \cdot \mathcal{L}_G^\ell(S)$, то от **(И.П.)** имаме, че
$$\alpha \in \{a^{n+1} b^m c^k \mid n + m \geq k\} \subseteq \{a^n b^m c^k \mid n + m \geq k\}.$$
- Ако $\alpha \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\}$, то от **(И.П.)** имаме, че
$$\alpha \in \{a^{n+1} b^m c^{k+1} \mid n + m \geq k\} \subseteq \{a^n b^m c^k \mid n + m \geq k\}.$$

За второто включване, нека $\alpha \in \mathcal{L}_G^{\ell+1}(B)$. Имаме три случая за думата α.

- Нека $\alpha \in \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\}$. Тогава от **(И.П.)** имаме, че:
$$\alpha \in \{b^{m+1} c^{k+1} \mid m \geq k\} \subseteq \{b^m c^k \mid m \geq k\}.$$
- Нека $\alpha \in \{b\} \cdot \mathcal{L}_G^\ell(B)$. Тогава от **(И.П.)** имаме, че:
$$\alpha \in \{b^{m+1} c^k \mid m \geq k\} \subseteq \{b^m c^k \mid m \geq k\}.$$
- Нека $\alpha \in \{\varepsilon\}$. Тогава е ясно, че имаме $\alpha \in \{b^m c^k \mid m \geq k\}$.

Най-накрая заключаваме, че

Така доказахме *коректност* на граматиката.

$$\mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) \subseteq \{a^n b^m c^k \mid n + m \geq k\}$$
$$\mathcal{L}_G(B) = \bigcup_\ell \mathcal{L}_G^\ell(B) \subseteq \{a^n b^m c^k \mid n + m \geq k\}.$$

Сега трябва да докажем обратните включвания, а именно:

Сега ще докажем *пълнота* на граматиката. Тук ще използваме представянията на $\mathcal{L}_G(S)$ и $\mathcal{L}_G(B)$ според *Следствие 3.2*.

$$\{a^n b^m c^k \mid n + m \geq k\} \subseteq \mathcal{L}_G(S) \tag{3.7}$$
$$\{b^m c^k \mid m \geq k\} \subseteq \mathcal{L}_G(B). \tag{3.8}$$

Трябва да започнем първо със *Свойство (3.8)*. Да разгледаме произволна дума $\alpha = b^m c^k$. Трябва да докажем, че $\alpha \in \mathcal{L}_G(B)$. Ще направим това с индукция по m.

- Нека $m = 0$. Това означава, че $\alpha = \varepsilon$. Ясно е, че $\varepsilon \in \mathcal{L}_G(B)$.
- Нека $m > 0$. Тук имаме два подслучая.
  - Нека $m = k$. Тогава $\alpha = b\gamma c$ и имаме, че $\gamma = b^{m-1} c^{k-1}$. Можем да приложим **(И.П.)** за γ и следователно $\gamma \in \mathcal{L}_G(B)$. Получаваме, че $\alpha \in \{b\} \cdot \mathcal{L}_G(B) \cdot \{c\} \subseteq \mathcal{L}_G(B)$.
  - Нека $m > k$. Тогава $\alpha = b\gamma$ и имаме, че $\gamma = b^{m-1} c^k$. Можем да приложим **(И.П.)** за γ и следователно $\gamma \in \mathcal{L}_G(B)$. Получаваме, че $\alpha \in \{b\} \cdot \mathcal{L}_G(B) \subseteq \mathcal{L}_G(B)$.

Сега преминаваме към *Свойство (3.7)*. Да разгледаме произволна дума $\alpha = a^n b^m c^k$. Трябва да докажем, че $\alpha \in \mathcal{L}_G(S)$. Ще направим това с индукция по n.

- Нека $n = 0$. Тогава $\alpha = b^m c^k$ и $m \geq k$. От *Свойство (3.8)* следва, че $\alpha \in \mathcal{L}_G(B) \subseteq \mathcal{L}_G(S)$.
- Нека $n > 0$. Имаме два подслучая.
  - Нека $n + m = k$. Тогава $\alpha = a\gamma c$ и $\gamma = a^{n-1} b^m c^{k-1}$. Можем да приложим **(И.П.)** за γ и следователно $\gamma \in \mathcal{L}_G(S)$. Получаваме, че $\alpha \in \{a\} \cdot \mathcal{L}_G(S) \cdot \{c\} \subseteq \mathcal{L}_G(S)$.
  - Нека $n + m > k$. Тогава $\alpha = a\gamma$ и $\gamma = a^{n-1} b^m c^k$. Можем да приложим **(И.П.)** за γ и следователно $\gamma \in \mathcal{L}_G(S)$. Получаваме, че $\alpha \in \{a\} \cdot \mathcal{L}_G(S) \subseteq \mathcal{L}_G(S)$.

Задача 3.4. Докажете, че езикът $L = \{a^n b^m c^k d^\ell \mid n + k = m + \ell\}$ е безконтекстен.

Упътване. Да разгледаме произволна дума от вида $\omega = a^n b^m c^k d^\ell$. Имаме два случая.

- Ако $n > \ell$, тогава имаме еквивалентността: $\omega \in L \Leftrightarrow (n - \ell) + k = m$.
- Ако $n \leq \ell$, тогава имаме еквивалентността: $\omega \in L \Leftrightarrow k = m + (\ell - n)$.

Това наблюдение ни подсказва, че е полезно да разгледаме следните езици:

$$L_1 = \{a^n b^m c^k \mid m = n + k\},$$
$$L_2 = \{b^m c^k d^\ell \mid k = m + \ell\}.$$

Така получаваме, че

$$L = \{a^n \cdot \omega \cdot d^n \mid n \in \mathbb{N} \,\&\, \omega \in L_1 \cup L_2\}.$$

L_1 е безконтекстен език, защото може да се опише с безконтекстната граматика G_1 със следните правила:

$$S_1 \to AC, \quad A \to aAb \mid \varepsilon, \quad C \to bCc \mid \varepsilon.$$

L_2 също е безконтекстен език, защото може да се опише с безконтекстната граматика G_2 със следните правила:

$$S_2 \to BD, \quad B \to bBc \mid \varepsilon, \quad D \to cCd \mid \varepsilon.$$

Тогава безконтекстната граматика G за L съдържа правилата на граматиките G_1 и G_2, а също и правилата

$$S \to aSd \mid S_1 \mid S_2.$$

□

Ние вече знаем, че този език не е регулярен

Задача 3.5. Докажете, че езикът $L = \{\alpha\beta \in \{a, b\}^\star \mid \ |\alpha| = |\beta| \,\&\, \alpha \neq \beta\}$ е безконтекстен.

Упътване. Да разгледаме една произволна дума ω, където $\omega = \alpha\beta$, $|\alpha| = |\beta|$ и $\alpha \neq \beta$. Знаем, че съществува индекс $i < |\alpha|$, такъв че думата ω може да се представи така:

$$\omega = \alpha[:i] \cdot \alpha[i] \cdot \alpha[i+1:] \cdot \beta[:i] \cdot \beta[i] \cdot \beta[i+1:],$$

където $\alpha[i] \neq \beta[i]$.

Нека $n = |\alpha| = |\beta|$ и да представим n като $n = i + 1 + k$. Имаме два случая.

- Ако $k \geq i$, то можем да преставим ω по следния начин:

$$\omega = \underbrace{\alpha[:i]}_{\text{дълж. } i} \cdot \alpha[i] \cdot \underbrace{\alpha[i+1:2i+1]}_{\text{дълж. } i} \cdot \underbrace{\alpha[2i+1:] \cdot \beta[:i]}_{\text{дълж. } k} \cdot \beta[i] \cdot \underbrace{\beta[i+1:]}_{\text{дълж. } k}.$$

- Нека $k < i$, то можем да преставим ω по следния начин:

$$\omega = \underbrace{\alpha[:i]}_{\text{дълж. } i} \cdot \alpha[i] \cdot \underbrace{\alpha[i+1:] \cdot \beta[:i-k]}_{\text{дълж. } i} \cdot \underbrace{\beta[i-k:i]}_{\text{дълж. } k} \cdot \beta[i] \cdot \underbrace{\beta[i+1:]}_{\text{дълж. } k}.$$

От тези представяния на ω е ясно, че можем да изразим езика L по следния начин:

$$L = L_a \cdot L_b \cup L_b \cdot L_a,$$

където:

$$L_a = \{\alpha \cdot a \cdot \beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta|\}$$
$$L_b = \{\alpha \cdot b \cdot \beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta|\}.$$

Сега разгледайте безконтекстната граматика G със следните правила:

$$S \to AB \mid BA$$
$$A \to XAX \mid a$$
$$B \to XBX \mid b$$
$$X \to a \mid b.$$

Лесно се съобразява, че $L_a = \mathcal{L}_G(A)$ и $L_b = \mathcal{L}_G(B)$ и накрая $L = \mathcal{L}_G(S)$. □

Задача 3.6. Докажете, че езикът $L = \{\alpha\sharp\beta \mid \alpha, \beta \in \{a,b\}^\star \ \&\ \alpha \neq \beta\}$ е безконтекстен.

Упътване. Разгледайте граматиката:

$$S \to AaR \mid BbR \mid E$$
$$A \to XAX \mid bR\sharp$$
$$B \to XBX \mid aR\sharp$$
$$E \to XEX \mid XR\sharp \mid \sharp XR$$
$$R \to XR \mid \varepsilon$$
$$X \to a \mid b.$$

Имаме, че за произволни думи $\alpha, \beta, \gamma, \delta \in \{a,b\}^\star$,

$$S \stackrel{\star}{\Rightarrow} \alpha b\gamma\sharp\beta a\delta \ \&\ |\alpha| = |\beta|,$$
$$S \stackrel{\star}{\Rightarrow} \alpha a\gamma\sharp\beta b\delta \ \&\ |\alpha| = |\beta|, \text{ или}$$
$$S \stackrel{\star}{\Rightarrow} \alpha\sharp\beta \ \&\ |\alpha| \neq |\beta|.$$

□

Задача 3.7. Докажете, че езикът $L = \{a^n b^m c^k d^\ell \mid n + k \geq m + \ell\}$ е безконтекстен.

Задача 3.8. Докажете, че езикът $L = \{\alpha\sharp\beta \mid \alpha, \beta \in \{a,b\}^\star \ \&\ |\alpha| \neq |\beta|\}$ е безконтекстен.

Задача 3.9. Да разгледаме граматиката G с правила

$$S \to AA \mid B,\ A \to B \mid bb,\ B \to aa \mid aB.$$

Да се намери езика на тази граматика и да се докаже, че граматиката разпознава точно този език.

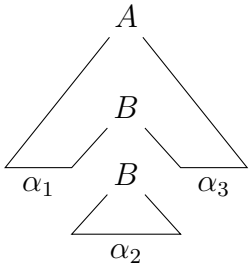
3.7 Лема за покачването

В този раздел ще разгледаме едно твърдение, което ще ни даде критерий за проверка кога един език не е безконтекстен. Ще започнем с няколко помощни твърдения.

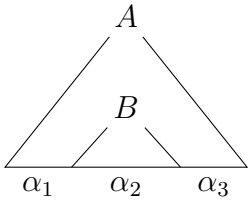
Твърдение 3.12. За произволни променливи A, B и думи $\alpha_1, \alpha_2, \alpha_3$ е изпълнено:

$$\frac{A \overset{\ell_1}{\lhd} \alpha_1 B \alpha_3 \qquad B \overset{\ell_2}{\lhd} \alpha_2}{A \overset{\leq \ell_1+\ell_2}{\lhd} \alpha_1\alpha_2\alpha_3}$$

Упътване. Формално доказателството протича с индукция по ℓ_1.



(а) Синтактични дървета за изводите $A \overset{\ell_1}{\lhd} \alpha_1 B \alpha_3$ и $B \overset{\ell_2}{\lhd} \alpha_2$



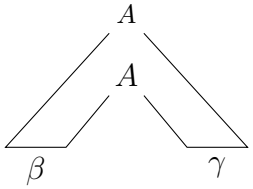
(б) Съединяваме двете дървета

□

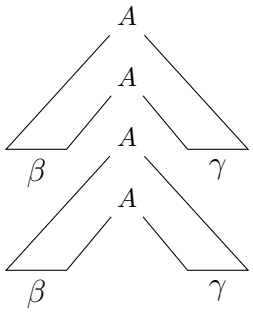
Твърдение 3.13. За произволна променлива A и произволни думи β и γ е изпълнено:

$$\frac{A \overset{\ell}{\lhd} \beta A \gamma \qquad i \in \mathbb{N}}{A \overset{\leq i \cdot \ell}{\lhd} \beta^i A \gamma^i}$$

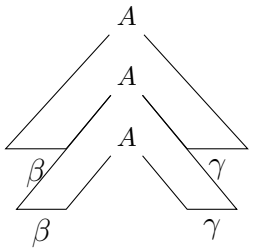
Упътване. Формално доказателството трябва да следва индукция по i. Понеже това доказателство не крие изненади, ще се задоволим с един пример. Нека P да бъде синтактично дърво съответстващо на извода $A \overset{\ell}{\lhd} \beta A \gamma$. Тогава можем да получим синтактично дърво $P^{(2)}$ съответстващо на $A \overset{\leq 2\ell}{\lhd} \beta^2 A \gamma^2$ по следния начин.



(а) Синтактично дърво P за извода $A \overset{\ell}{\lhd} \beta A \gamma$



(б) Съединяваме две копия на P



(в) Получаваме синтактично дърво $P^{(2)}$ за извода $A \overset{\leq 2\ell}{\lhd} \beta^2 A \gamma^2$

□

Твърдение 3.14. Нека G е безконтекстна граматика и

$$b \stackrel{\text{деф}}{=} \max\{\, |\gamma| \mid A \to \gamma \text{ е правило в } G \,\}.$$

Числото b е максималната разклоненост, която можем да получим за синтактично дърво на извод в граматиката G.

Тогава:

- Ако $A \overset{\leq \ell}{\lhd} \alpha$, то $|\alpha| \leq b^\ell$.
- Ако $\alpha \in \mathcal{L}(G)$ и $|\alpha| > b^\ell$, то $S \overset{\geq \ell+1}{\lhd} \alpha$.

Доказателство.

Това твърдение е на практика следствие от *Задача 1.13* в комбинация с *Твърдение 3.10*.

- Да разгледаме синтактично дърво $P = (T, \lambda)$ за извода $A \overset{\leq \ell}{\lhd} \alpha$. Според *Задача 1.13* имаме, че $|\alpha| = |\texttt{leaves}(T)| \leq b^{\texttt{height}(T)} = b^\ell$.
- Нека $\alpha \in \mathcal{L}(G)$ и $|\alpha| > b^\ell$. Това означава, че $S \overset{\star}{\lhd} \alpha$. Ако допуснем, че $S \overset{\leq \ell}{\lhd} \alpha$, то бихме имали, че $|\alpha| \leq b^\ell$, което е противоречие. Заключаваме, че $S \overset{\geq \ell+1}{\lhd} \alpha$.

□

Лема 3.5 (за покачването (безконтекстни езици)). За всеки безконтекстен език L съществува $p > 0$, такова че ако $\alpha \in L$, за която $|\alpha| \geq p$, то съществува разбиване на думата на пет части, $\alpha = xyuvw$, за което е изпълнено:

1) $|yv| \geq 1$,
2) $|yuv| \leq p$, и
3) $(\forall i \in \mathbb{N})[xy^iuv^iw \in L]$.

[Sip12, стр. 125], [HU79, стр. 125], [HMU01, стр. 275], [Koz97, стр. 148].
Ще казваме, че p е константа на покачването. Тук отново да напомним, че $0 \in \mathbb{N}$ и $xy^0uv^0w = xuw$.

Доказателство. Нека G е безконтекстна граматика за езика L. Да положим

$$p \stackrel{\text{деф}}{=} b^{|V|+1}, \text{ където } b \stackrel{\text{деф}}{=} \max\{\, |\gamma| \mid A \to \gamma \text{ е правило в } G \,\}.$$

Числото b е максималната разклоненост на всяко дърво на извод за дума изводима от граматиката G.

Ще покажем, че този избор на p е удачен за удовлетворяването на условията на лемата. Ще наричаме p константа на покачването за граматиката G. Да разгледаме произволна дума $\alpha \in \mathcal{L}(G)$, за която $|\alpha| \geq p$. Понеже $\mathcal{L}(G)$ е безкраен език, то със сигурност можем да намерим такава дума. Щом $S \overset{\star}{\lhd} \alpha$, според *Твърдение 3.14*, винаги имаме $S \overset{\ell}{\lhd} \alpha$ за $\ell \geq |V| + 1$. Нека измежду всички синтактични дървета

Важното е да вземем дума $\alpha \in \mathcal{L}(G)$, за която $|\alpha| > b^{|V|}$.

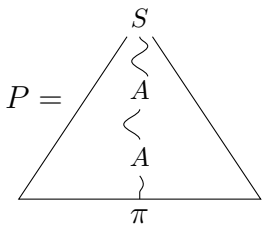
$P = (T, \lambda)$ за извода $S \overset{\ell}{\lhd} \alpha$ сме избрали такова с *минимален* брой елементи в T. Да фиксираме *максимален* път π в T, т.е. дума $\pi \in T$ и $|\pi| = \ell$. Чрез функцията λ пътя π определя редицата $X_0, X_1, \ldots, X_\ell$, като $X_i = \lambda(\rho)$, където $\rho \prec \pi$ и $|\rho| = i$. Ясно е, че $X_i \in V$ за $i < \ell$ и $X_\ell \in \Sigma \cup \{\varepsilon\}$. Щом $\ell \geq |V| + 1$, то това означава, че по този път π се срещат $\ell \geq |V| + 1$ на брой променливи. От принципа на Дирихле следва, че трябва да има повторения на променливи по този път. Във вече фиксираното синтактично дърво P можем да изберем това срещане на двойка повтарящи се променливи по пътя π, както на *Фигура 3.8а*, което е възможно най-близо до края на пътя. Това означава, че можем да разбием извода $S \overset{\ell}{\lhd} \alpha$ като $S \overset{\star}{\lhd} xAw$ и $A \overset{\leq |V|+1}{\lhd} \gamma$, където $\alpha = x\gamma w$. Сега според нашия избор, изводът $A \overset{\leq |V|+1}{\lhd} \gamma$ може да се разбие като $A \overset{\leq |V|+1}{\lhd} yAv$ и $A \overset{\leq |V|}{\lhd} u$, където $\gamma = yuv$. Да обобщим. Можем да разбием думата α на пет части като $\alpha = xyuvw$ с изводите:

Принципът на Дирихле е известен също и като принципа на чекмеджетата.

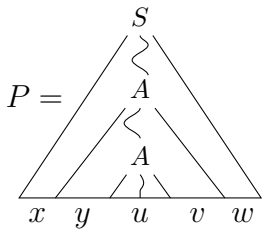
(1) $S \overset{\star}{\lhd} xAw$,

(2) $A \overset{\leq |V|+1}{\lhd} yuv$, защото в дървото P можем да изберем първата двойка повтарящи се променливи, които срещнем отзад напред по пътя π. Тук сме означили тази повтаряща се променлива с .

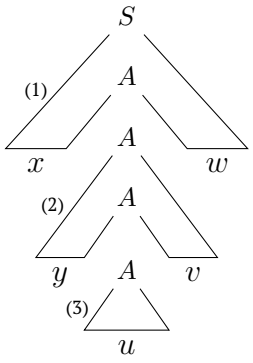
(3) $A \overset{\leq |V|}{\lhd} u$, като тук вече няма повтарящи се променливи по останалата част от пътя π.



(а) Първата двойка повтарящи се променливи, които намерим като тръгнем отдолу нагоре по пътя π.



(б) Разбиваме дървото на три части.



(в) Получаваме три отделни синтактични дървета.

Сега остава да проверим условието на лемата:

- Да допуснем, че $|yv| = 0$. Това означава, че $\alpha = xuw$ и имаме извода:

$$\frac{S \overset{\star}{\lhd} xAw \qquad A \overset{\star}{\lhd} u}{S \overset{\star}{\lhd} \underbrace{xuw}_{\alpha},} \text{ (Твърдение 3.12)}$$

за който съществува дърво на извод с по-малко на брой възли отколкото P, защото махаме средната част, която съдържа поне един възел. Това е противоречие с избора на P като синтактично дърво за $S \overset{\star}{\lhd} \alpha$ с минимален брой възли. Заключаваме, че $|yv| \geq 1$.

- Понеже имаме извода $A \overset{\leq |V|+1}{\lhd} yuv$, то от *Твърдение 3.14* следва, че $|yuv| \leq b^{|V|+1} = p$.
- За произволно $i \in \mathbb{N}$ имаме извода:

$$\dfrac{\dfrac{\dfrac{A \overset{\star}{\lhd} yAv}{A \overset{\star}{\lhd} y^iAv^i}\ (\text{Твърдение 3.13}) \qquad A \overset{\star}{\lhd} u}{A \overset{\star}{\lhd} y^iuv^i}\ (\text{Твърдение 3.12}) \qquad S \overset{\star}{\lhd} xAw}{S \overset{\star}{\lhd} xy^iuv^iw}\ (\text{Твърдение 3.12})$$

Оттук заключаваме, че $(\forall i \in \mathbb{N})[xy^iuv^iw \in \mathcal{L}(G)]$.

□

Както и при Лема за покачването за регулярни езици, ние ще използваме *контрапозицията* на условието на Лема за покачването за безконтекстни езици при проверка, че даден език не е безконтекстен. [Koz97, стр. 153].

Следствие 3.3. Нека L е произволен **безкраен** език. Нека също така е изпълнено, че: Ако L е краен език, то е ясно, че L е безконтекстен.

(∀) за *всяко* естествено число $p \geq 1$,

(∃) можем да намерим дума $\alpha \in L$, $|\alpha| \geq p$, такава че

(∀) за *всяко* разбиване на думата на пет части, $\alpha = xyuvw$, със свойствата $|yv| \geq 1$ и $|yuv| \leq p$,

(∃) можем да намерим $i \in \mathbb{N}$, за което е изпълнено, че $xy^iuv^iw \notin L$.

Тогава L **не** е безконтекстен език. ✍ Докажете! Аналогично е на *Следствие 2.1*

Следствие 3.4. Нека G е безконтекстна граматика и p е константата на покачването за G. Тогава $|\mathcal{L}(G)| = \infty$ точно тогава, когато съществува $\alpha \in \mathcal{L}(G)$, за която $p \leq |\alpha| < 2p$. ✍ Докажете!

Забележка. Алгоритъм за проверка дали един безконтекстен език е безкраен следвайки горния критерий би имал експоненциална сложност относно $|G|$. ✍ Защо?

Пример 3.9. Да видим защо езикът $L = \{a^nb^nc^n \mid n \in \mathbb{N}\}$ не е безконтекстен.

(∀) Разглеждаме произволна константа $p \geq 1$.

($\exists$) Избираме дума $\alpha \in L$, $|\alpha| \geq p$. В случая, нека $\alpha = a^p b^p c^p$.

($\forall$) Разглеждаме произволно разбиване на α на пет части $\alpha = xyuvw$, за което $|yuv| \leq p$ и $1 \leq |yv|$.

✍ Съобразете, че може да изберете и $i = 0$. В този пример може да вземете едно и също i за всяко разбиване на думата α от предишната стъпка. В други примери ще видим, че изборът на i зависи от разглежданото разбиване на думата.

($\exists$) За всяко такова разбиване ще посочим i, за което $xy^i uv^i w \notin L$. Знаем, че поне едно от y и v не е празната дума. Имаме няколко случая за y и v.

- y и v са думи съставени от една буква. В този случай получаваме, че xy^2uv^2w има различен брой букви a, b и c.
- y или v е съставена от две букви. В този случай получаваме, че в xy^2uv^2w редът на буквите е нарушен.
- понеже $|yuv| \leq p$, то не е възможно в y или v да се срещат и трите букви.

Оказа се, че във всички възможни случаи за y и v, $xy^2uv^2w \notin L$.

Така от *Следствие 3.3* следва, че езикът L не е безконтекстен.

Твърдение 3.15. Безконтекстните езици **не** са затворени относно операциите сечение и допълнение.

Упътване. Да разгледаме езика $L_0 = \{a^n b^n c^n \mid n \in \mathbb{N}\}$, за който вече знаем от *Пример 3.9*, че не е безконтекстен. Да вземем също така и безконтекстните езици

✍ Защо L_1 и L_2 са безконтекстни?

$$L_1 = \{a^n b^n c^m \mid n, m \in \mathbb{N}\},\ L_2 = \{a^m b^n c^n \mid n, m \in \mathbb{N}\}.$$

Да напомним, че с $\overline{L}$ означаваме допълнението на езика L, т.е. $\overline{L} \stackrel{\text{деф}}{=} \Sigma^\star \setminus L$.

- Понеже $L_0 = L_1 \cap L_2$, то заключаваме, че безконтекстните езици не са затворени относно операцията сечение.

Друг пример, с който може да се види, че безконтекстните езици не са затворени относно допълнение е като се докаже, че езикът

$\{a, b\}^\star \setminus \{\omega\omega \mid \omega \in \{a, b\}^\star\}$

е безконтекстен. Това следва лесно като се използва *Задача 3.5*.

- Да допуснем, че безконтекстните езици са затворени относно операцията допълнение. Тогава $\overline{L}_1$ и $\overline{L}_2$ са безконтекстни. Знаем, че безконтекстните езици са затворени относно обединение. Следователно, езикът $L_3 = \overline{L}_1 \cup \overline{L}_2$ също е безконтекстен. Понеже допуснахме, че безконтекстните са затворени относно допълнение, то $\overline{L}_3$ също е безконтекстен. Но тогава получаваме, че езикът

$$L_0 = L_1 \cap L_2 = \overline{\overline{L}_1 \cup \overline{L}_2} = \overline{L}_3$$

е безконтекстен, което е противоречие.

□

Примерни задачи

Задача 3.10. Докажете, че езикът $L = \{a^i b^j c^k \mid 0 \leq i \leq j \leq k\}$ не е безконтекстен.

Упътване. Прилагаме *Следствие 3.3*.

($\forall$) Разглеждаме произволна константа $p \geq 1$.

($\exists$) Избираме конкретна дума $\alpha \in L$, $|\alpha| \geq p$. В случая, нека $\alpha = a^p b^p c^p$.

($\forall$) Разглеждаме произволно разбиване $xyuvw = \alpha$, за което $|yuv| \leq p$ и $1 \leq |yv|$. Знаем, че поне една от y и v не е празната дума.

($\exists$) Ще посочим конкретно $i \in \mathbb{N}$, за което $xy^i uv^i w \notin L$.

Обърнете внимание, че тук i съществено зависи от вида на разбиването.

- y и v са съставени от една буква. Имаме три случая.
  - i) a не се среща в y и v. Тогава xy^0vu^0w съдържа повече a от b или c.
  - ii) b не се среща в y и v.
    - Ако a се среща в y или v, тогава xy^2uv^2w съдържа повече a от b.
    - Ако c се среща в y или v, тогава xy^0uv^0w съдържа по-малко c от b.
  - iii) c не се среща в y и v. Тогава xy^2uv^2w съдържа повече a или b от c.
- y или v е съставена от две букви. Тук разглеждаме xy^2uv^2w и съобразяваме, че редът на буквите е нарушен.

□

Задача 3.11. Докажете, че езикът $L = \{\, \alpha\alpha \mid \alpha \in \{a,b\}^\star \,\}$ не е безконтекстен.

Упътване.

- Защо $\omega = a^p b a^p b$ не става ?
- Защо $\omega = a^p b^{2p} a^p$ не става ?
- Разгледайте $\omega = a^p b^p a^p b^p$.

□

Задача 3.12. Докажете, че езикът $L = \{\alpha\beta\alpha^{\mathtt{rev}} \mid \alpha, \beta \in \{a,b\}^\star \;\&\; |\alpha| = |\beta|\}$ не е безконтекстен.

Упътване.

- Защо не става ако разгледаме думата $\alpha = a^p b^p a^p$?
- Защо не става ако разгледаме думата $\alpha = a^p b^p a^{2p} b^p a^p$?
- Разгледайте $\alpha = a^p b^p a^p b^p b^p a^p$. Покачване с повече от p би трябвало да свърши работа.

□

Задача 3.13. Докажете, че езикът $L = \{\alpha\beta\alpha \mid \alpha, \beta \in \{a,b\}^\star \;\&\; |\alpha| \geq 1\}$ не е безконтекстен.

Упътване.

- Защо не става с $\omega = a^p b a^p b$?
- Защо не става с $\omega = a b^p a b^p$?
- Пробвайте с $\omega = a^p b^p a^p b^p$.

□

Задача 3.14. Докажете, че езикът $L = \{\alpha\sharp\beta \mid \alpha \text{ е подниз на } \beta\}$ не е безконтекстен.

Упътване.

- Защо не става ако вземем $\omega = a^p\sharp a^p$?
- Защо не става ако вземем $\omega = a^pb\sharp a^pb$?
- Разгледайте $\omega = a^pb^p\sharp a^pb^p$.

□

Задача 3.15. Вярно ли е, че следните езици са безконтекстни:

а) $L = \{\alpha\sharp\beta \mid \alpha, \beta \in \{0,1\}^\star \,\&\, \overline{\alpha}_{(2)} + 1 = \overline{\beta}_{(2)}\}$;

б) $L = \{\alpha\sharp\beta^{\mathtt{rev}} \mid \alpha, \beta \in \{0,1\}^\star \,\&\, \overline{\alpha}_{(2)} + 1 = \overline{\beta}_{(2)}\}$?

3.8 Алгоритми

Тук ще докажем, че съществуват *полиномиални* алгоритми, които за дадена безконтекстна граматика G проверяват:

- дали $\mathcal{L}(G) = \emptyset$;
- дали $|\mathcal{L}(G)| = \infty$;
- дали $|\mathcal{L}(G)| < \infty$;
- по дадена дума α дали $\alpha \in \mathcal{L}(G)$.

Нека приемем, че дължината на граматика G е

$$|G| \stackrel{\text{деф}}{=} |V| + |\Sigma| + \sum_{(A,\alpha)\in R} |A\alpha|.$$

3.8.1 Опростяване на безконтекстни граматики

Премахване на безполезните променливи

Нека е дадена безконтекстната граматика $G = \langle V, \Sigma, R, S\rangle$. Една променлива A се нарича **полезна**, ако съществува извод от следния вид:

[HU79, стр. 88], [HMU01, стр. 256].

$$S \stackrel{\star}{\Rightarrow}_G \alpha A \beta \stackrel{\star}{\Rightarrow}_G \gamma,$$

където $\gamma \in \Sigma^\star$, а $\alpha, \beta \in (V \cup \Sigma)^\star$. Това означава, че една променлива е полезна, ако участва в извода на някоя дума в езика на граматиката. Една променлива се нарича **безполезна**, ако не е полезна. Целта ни е да получим еквивалентна граматика G' без безполезни променливи. Ще решим задачата като разгледаме две леми.

Лема 3.6. Нека е дадена безконтекстната граматика G и $\mathcal{L}(G) \neq \emptyset$. Съществува алгоритъм, който намира граматика G', за която $\mathcal{L}(G) = \mathcal{L}(G')$ и със свойството, че за всяка променлива $A' \in V'$, съществува дума $\alpha \in \Sigma^\star$, за която $A' \stackrel{\star}{\Rightarrow}_{G'} \alpha$.

Упътване. Целта ни е да намерим множеството `Gen` от променливи, които генерират думи, т.е. търсим множеството

$$\texttt{Gen} \stackrel{\text{деф}}{=} \{A \in V \mid A \stackrel{\star}{\lhd} \alpha \,\&\, \alpha \in \Sigma^\star\}.$$

За целта ще построим редица от множества `Gen[n]` по следния начин:

Всяка итерация на алгоритъма отнема $\mathcal{O}(|G|)$ време. Следователно, изпълнението на целия алгоритъм отнема $\mathcal{O}(|G|^2)$ време.

- `Gen[0]` $\stackrel{\text{деф}}{=} \emptyset$;
- `Gen[n+1]` $\stackrel{\text{деф}}{=} \{A \in V \mid (\exists \alpha \in (\Sigma \cup$ `Gen[n]` $)^\star)[A \to_G \alpha]\}$.
- Спираме, когато стигнем до такова n, за което `Gen[n]` $=$ `Gen[n+1]`. Лесно се съобразява, че $(\forall k \geq n)[$`Gen[n]` $=$ `Gen[k]`$]$.

✍ Докажете сами!

Трябва да докажем, че $\texttt{Gen} = \texttt{Gen[n]}$. Това ще направим като докажем, че за всяко k,

$$A \in \texttt{Gen[k]} \Leftrightarrow (\exists \alpha \in \Sigma^\star)[A \overset{\leq k}{\lhd} \alpha]$$

Дефинираме G' като $V' = \texttt{Gen[n]}$ и правилата на G' са само тези правила на G, в които участват променливи от V' и букви от Σ. □

Следствие 3.5. Съществува *полиномиален* алгоритъм, който определя за всяка безконтекстна граматика G дали $\mathcal{L}(G) = \emptyset$.

Доказателство. Прилагаме алгоритъма от *Лема 3.6* и намираме множеството от променливи V'. Тогава $\mathcal{L}(G) = \emptyset$ точно тогава, когато $S \notin V'$. □

Лема 3.7. Съществува алгоритъм, който по дадена безконтекстна граматика $G = \langle V, \Sigma, R, S\rangle$, намира $G' = \langle V', \Sigma, S, R'\rangle$, $\mathcal{L}(G') = \mathcal{L}(G)$, със свойството, че за всяко $X \in V'$ съществуват $\alpha, \beta \in (V' \cup \Sigma)^\star$, за които $S \overset{\star}{\Rightarrow} \alpha X \beta$, т.е. всяка променлива в G' е достижима от началната променлива S.

Упътване. Нашата цел е да намерим множеството

$$\texttt{Reach} \overset{\text{деф}}{=} \{B \in V \mid S \overset{\star}{\lhd} \alpha B \beta, \text{ за някои } \alpha, \beta \in (\Sigma \cup V)^\star\}.$$

Новата граматика G' ще има променливи $V' \overset{\text{деф}}{=} \texttt{Reach}$, като правилата на граматика G' са същите като тези на G, но ограничени до V'. Лесно се доказава, че $\mathcal{L}(G) = \mathcal{L}(G')$. Остава да намерим множеството $\texttt{Reach}$. За да постигнем тази цел, ще започнем да строим множествата $\texttt{Reach[i]} \subseteq V$ по следния начин:

$$\texttt{Reach[0]} \overset{\text{деф}}{=} \{S\}$$

$$\texttt{Reach[i+1]} \overset{\text{деф}}{=} \{B \in V \mid (\exists A \in \texttt{Reach[i]})[A \overset{\leq 1}{\lhd} \alpha B \beta, \text{ за някои } \alpha, \beta \in (V \cup \Sigma)^\star]\},$$

което може да се запише и така:

$$\texttt{Reach[i+1]} = \{B \in V \mid (\exists A \in \texttt{Reach[i]})[A \to_G \alpha B \beta, \text{ за някои } \alpha, \beta \in (V \cup \Sigma)^\star]\} \cup \texttt{Reach[i]}.$$

Докажете, че за всяко i е изпълнено:

$$\texttt{Reach[i]} = \{B \in V \mid S \overset{\leq i}{\lhd} \alpha B \beta, \text{ за някои } \alpha, \beta \in (\Sigma \cup V)^\star\}.$$

✍ Защо сме сигурни, че ще достигнем такава стъпка?

Спираме да строим тези множества, когато достигнем до стъпка n, за която

$$\texttt{Reach[n]} = \texttt{Reach[n+1]}.$$

Съобразете, че за това n е изпълнено, че $\texttt{Reach[n]} = \texttt{Reach}$, т.е.

$$\texttt{Reach[n]} = \{B \in V \mid S \overset{\star}{\lhd} \alpha B \beta, \text{ за някои } \alpha, \beta \in (\Sigma \cup V)^\star\}.$$

□

Пример 3.10. Да разгледаме безконтекстната граматика G с правила:

$$S \to AB \mid aA$$
$$A \to a \mid aAa$$
$$B \to SB \mid BC$$
$$C \to \varepsilon \mid cC.$$

Първо да намерим променливите, от които се извеждат думи.

- $\texttt{Gen[0]} = \emptyset$;
- $\texttt{Gen[1]} = \{A, C\}$, защото $A \to a$ и $C \to \varepsilon$;
- $\texttt{Gen[2]} = \{A, C, S\}$, защото $S \to aA$;
- не можем да добавим B към `Gen[3]`, следователно $\texttt{Gen[3]} = \texttt{Gen[2]}$.

Получаваме граматиката G':

$$S \to aA$$
$$A \to a \mid aAa$$
$$C \to \varepsilon \mid cC.$$

Сега премахваме променливите и буквите, които не са достижими от началната промелива S.

- $\texttt{Reach[0]} = \{S\}$;
- $\texttt{Reach[1]} = \{S, A\}$, защото $S \to aAa$;
- $\texttt{Reach[2]} = \{S, A\}$.

Така получаваме граматиката G'':

$$S \to aA$$
$$A \to a \mid aAa.$$

Теорема 3.4. За всяка безконтекстна граматика G, за която $\mathcal{L}(G) \neq \emptyset$, съществува еквивалетнтна на нея безконтекстна граматика G' без безполезни правила. Освен това, граматиката G' може да се намери за полиномиално време.

Упътване. Прилагаме върху G първо процедурата от *Лема 3.6* и след това върху резултата прилагаме процедурата от *Лема 3.7*. □

✍ Защо е важна последователността на прилагане? Например, да разгледаме граматиката

$$S \to AB \mid ab$$
$$A \to aA \mid \varepsilon$$
$$B \to bB.$$

Ако първо приложим процедурата от *Лема 3.7*, то нищо няма да премахнем, защото A и B са достижими от S. След това, ако приложим процедурата от *Лема 3.6*, то ще премахнем всички правила, в които участва B. Така A става недостижима от S и трябва пак да приложим процедурата от *Лема 3.7*. Алгоритъмът описан тук е квадратичен. Има и линеен такъв. Вижте [HMU01, стр. 296]

Задача 3.16. Проверете дали $\mathcal{L}(G) = \emptyset$, където правилата на G са:

$$S \to AS \mid BC$$
$$A \to a \mid BA \mid SB$$
$$B \to b \mid BC \mid AB$$
$$C \to CB \mid SC \mid AS.$$

Премахване на ε-правила

Лема 3.8. Съществува *експоненциален* алгоритъм, такъв че превръща всяка безконтекстна граматика G в безконтекстна граматика G' без правила от вида $A \to \varepsilon$, и $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

Упътване. За да премахнем правилата от вида $A \to \varepsilon$ от граматиката, първо трябва да намерим множеството от променливи

$$\texttt{E} = \{A \in V \mid A \overset{\star}{\lhd} \varepsilon\}.$$

За да направим това, следваме процедурата:

1) Първо строим множествата `E[n]` по следния начин:

   - $\texttt{E[0]} \overset{\text{деф}}{=} \emptyset$;

На всяка стъпка трябва да обходим цялата граматика, следователно всяка итерация отнема $\mathcal{O}(|G|)$ време.

Обърнете внимание, че имаме $\texttt{E[n]}^\star$, а не просто $\texttt{E[n]}$. Също така да напомним, че $\emptyset^\star = \{\varepsilon\}$.

- $\texttt{E[n+1]} \stackrel{\text{деф}}{=} \{B \in V \mid (\exists \alpha \in \texttt{E[n]}^\star)[B \to_G \alpha]\}$.
- Докажете, че за всяко n е изпълнено, че

$$\texttt{E[n]} = \{A \in V \mid A \overset{\leq n}{\lhd} \varepsilon\}.$$

Намираме E за време $\mathcal{O}(|G|^2)$.

- Спираме на първото n, за което $\texttt{E[n]} = \texttt{E[n+1]}$. Лесно се съобразява, че за това n е изпълнено, че

$$\texttt{E[n]} = \{A \in V \mid A \overset{\star}{\lhd} \varepsilon\}.$$

2) Строим множеството от правила R', в което няма ε-правила по следния начин. За всяко правило $A \to_G X_1 \cdots X_k$, добавяме към R' всички правила от вида $A \to_G Y_1 \cdots Y_k$, където:

Броят на правилата може да се увеличи експоненциално, защото в най-лошия случай извеждаме всички подмножества на дадено множество от променливи

- ако $X_i \notin \texttt{E}$, то $Y_i = X_i$;
- ако $X_i \in \texttt{E}$, то $Y_i = X_i$ или $Y_i = \varepsilon$;
- не всички Y_i са ε.

✍ Докажете сами!

Лесно се съобразява, че $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

□

Пример 3.11. Нека е дадена граматиката G с правила

$$\begin{aligned} S &\to D \\ D &\to AD \mid b \\ A &\to AB \mid BC \mid a \\ B &\to AA \mid UC \\ C &\to \varepsilon \mid CA \mid a \\ U &\to \varepsilon \mid aUb. \end{aligned}$$

Тогава

- $\texttt{E[0]} = \emptyset$;
- $\texttt{E[1]} = \{C, U\}$;
- $\texttt{E[2]} = \{C, U, B\}$;
- $\texttt{E[3]} = \{C, U, B, A\}$;
- $\texttt{E[4]} = \texttt{E[3]}$. Следователно,

$$\texttt{E} = \{X \in V \mid X \overset{\star}{\lhd} \varepsilon\} = \{C, U, B, A\}.$$

Това означава, че $\varepsilon \notin \mathcal{L}(G)$. Граматиката G' без ε-правила, за която $\mathcal{L}(G') = \mathcal{L}(G)$ има следните правила

$$\begin{aligned} S &\to D \\ D &\to AD \mid D \mid b \\ A &\to A \mid B \mid C \mid AB \mid BC \mid a \\ B &\to A \mid C \mid AA \mid U \mid UC \\ C &\to C \mid A \mid CA \mid a \\ U &\to aUb \mid ab. \end{aligned}$$

Премахване на преименуващи правила

В [HMU01, стр. 263] преименуващите правила се наричат *unit productions*.

Едно правило в граматиката G се нарича **преименуващо**, ако е от вида $A \to B$. Нека е дадена граматика $G = \langle V, \Sigma, R, S \rangle$. Ще построим еквивалентна граматика G' без преименуващи правила. В началото нека в G' да добавим всички правила от G, които не са преименуващи. След това, за всяка променлива A, за която $A \stackrel{\star}{\Rightarrow}_G B$, ако $B \to \alpha$ е правило в граматиката G, което не е преименуващо, то добавяме към G' правилото $A \to \alpha$.

Лема 3.9. Нека G е безконтекстна граматика без ε-правила. Съществува *полиномиален* алгоритъм, такъв че превръща всяка безконтекстна граматика G в безконтекстна граматика G' без преименуващи правила и $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

Упътване. Първо намираме множеството от двойки

$$\texttt{Ren} = \{(A, B) \in V \times V \mid A \stackrel{\star}{\Rightarrow} B\}$$

като строим множествата по такъв начин, че да е изпълнено свойството:

$$\texttt{Ren[n]} = \{(A, B) \in V \times V \mid A \stackrel{\leqslant n}{\Rightarrow} B\}$$

$$\texttt{Ren[0]} \stackrel{\text{деф}}{=} \{(A, A) \mid A \in V\}$$
$$\texttt{Ren[n+1]} \stackrel{\text{деф}}{=} \texttt{Ren[n]} \cup \{(A, B) \mid (\exists C)[A \to_G C \,\&\, (C, B) \in \texttt{Ren[n]}]\}.$$

Спираме на първото n, за което $\texttt{Ren[n]} = \texttt{Ren[n+1]}$. Тогава $\texttt{Ren} = \texttt{Ren[n]}$. Нека $R'_0 \stackrel{\text{деф}}{=} R \setminus (V \times V)$ са правилата на R, които не са преименуващи. Тогава

$$R' \stackrel{\text{деф}}{=} \{\, (A, \alpha) \in V \times (V \cup \Sigma)^\star \mid (\exists B)[(A, B) \in \texttt{Ren} \,\&\, (B, \alpha) \in R'_0] \,\}.$$

|Ren| има големина $\mathcal{O}(|G|^2)$.

□

Пример 3.12. Нека е дадена граматиката G с правила

$$S \to B \mid CC \mid b$$
$$A \to S \mid SB$$
$$B \to C \mid BC$$
$$C \to AB \mid a \mid b.$$

Да намерим първо множеството Ren.

$$\begin{aligned}
\texttt{Ren[0]} &= \{(S,S),(A,A),(B,B),(C,C)\};\\
\texttt{Ren[1]} &= \{(S,S),(S,B),(A,A),(A,S),\\
&\quad (B,B),(B,C),(C,C)\};\\
\texttt{Ren[2]} &= \{(S,S),(S,B),(S,C),(B,B),\\
&\quad (B,C),(C,C),(A,A),(A,S),\\
&\quad (A,B)\};\\
\texttt{Ren[3]} &= \{(S,S),(S,B),(S,C),(B,B),\\
&\quad (B,C),(C,C),(A,A),(A,S),\\
&\quad (A,B),(A,C)\};\\
\texttt{Ren[4]} &= \{(S,S),(S,B),(S,C),(B,B),\\
&\quad (B,C),(C,C),(A,A),(A,S),\\
&\quad (A,B),(A,C)\}.
\end{aligned}$$

Получихме, че $\texttt{Ren[3]} = \texttt{Ren[4]}$. Оттук можем да заключим следното:

$$A \stackrel{\star}{\Rightarrow} A, B, S, C$$
$$B \stackrel{\star}{\Rightarrow} B, C$$
$$S \stackrel{\star}{\Rightarrow} S, B, C$$
$$C \stackrel{\star}{\Rightarrow} C.$$

Първо добавяме към R' правилата, които не са преименуващи, а именно:

$$A \to SB$$
$$B \to BC$$
$$C \to AB \mid a \mid b$$
$$S \to CC \mid b.$$

- Понеже имаме,че $A \stackrel{\star}{\Rightarrow} B, S, C$, то добавяме към R' правилата:

$$A \to BC \mid AB \mid a \mid b \mid CC.$$

- Понеже имаме,че $B \stackrel{\star}{\Rightarrow}_G C$, то добавяме към R' правилата:

$$B \to AB \mid a \mid b.$$

- Понеже имаме, че $S \stackrel{\star}{\Rightarrow}_G B, C$, то добавяме към R' правилата:

$$S \to BC \mid AB \mid a \mid b.$$

Накрая получаваме, че граматиката G' има правила

$$\begin{aligned} S &\to BC \mid AB \mid CC \mid a \mid b \\ A &\to BS \mid BC \mid AB \mid a \mid b \mid CC \\ B &\to AB \mid a \mid b \mid BC \\ C &\to AB \mid a \mid b. \end{aligned}$$

Задача 3.17. Премахнете преименуващите правила от граматиката G, като запазите езика, ако G има следните правила:

$$\begin{aligned} S &\to C \mid CC \mid b \\ A &\to B \\ B &\to S \mid C \mid BC \\ C &\to a \mid AB; \end{aligned}$$

Премахване на дългите правила

Новата граматика ще има дължина $\mathcal{O}(|G|)$.

Едно правило се нарича дълго, ако е от вида $A \to \beta$, където $|\beta| \geq 3$. Да разгледаме едно дълго правило в граматиката от вида $A \to X_1 X_2 \cdots X_k$, където $k \geq 3$. За да получим еквивалентна граматика без това дълго правило, добавяме нови променливи $A_1, \ldots, A_{k-2}$, и правила

$$A \to X_1 A_1,\ A_1 \to X_2 A_2, \ldots,\ A_{k-2} \to X_{k-1} X_k.$$

Задача 3.18. Нека е дадена граматиката $G = \langle \{S, A, B, C\}, \{a, b\}, S, R \rangle$. Използвайте обща конструкция, за да премахнете „дългите" правила от G като при това получите безконтестна граматика G_1 с език $\mathcal{L}(G) = \mathcal{L}(G_1)$, където правилата на граматиката са:

$$\begin{aligned} S &\to CC \mid b \\ A &\to BSB \mid a \\ B &\to ba \mid BC \\ C &\to BaSA \mid a \mid b. \end{aligned}$$

3.8.2 Нормална Форма на Чомски

Една безконтекстна граматика е в **нормална форма на Чомски**, ако всяко правило е от вида

$$A \to BC \text{ и } A \to a,$$

Ако искаме ε да бъде в езика на граматиката, то позволяваме от началната променлива S да имаме правилото $S \to_G \varepsilon$, но тогава забраняваме S да се среща в дясна страна на правило.

където A, B, C са произволни променливи и a е произволна буква.

Теорема 3.5. Всеки безконтекстен език L, който не съдържа ε, се поражда от безконтекстна граматика в нормална форма на Чомски.

Доказателство. Нека имаме безконтекстна граматика G, за която $L = \mathcal{L}(G)$. Ще построим безконтекстна граматика G' в нормална форма на Чомски, $L = \mathcal{L}(G')$. Следваме следната процедура:

- Премахваме дългите правила. Това можем да направим за време $\mathcal{O}(|G|)$ като новата граматика ще има дължина $\mathcal{O}(|G|)$.
- Премахваме ε-правилата. Това можем да направим за време $\mathcal{O}(|G|^2)$ като новата граматика ще има дължина $\mathcal{O}(|G|)$.

Редът на стъпките е важен. Трябва преди това сме премахнали дългите правила. Вижте [HMU01, стр. 296].

- Премахваме преименуващите правила. Това можем да направим за време $\mathcal{O}(|G|^2)$ като новата граматика ще има дължина $\mathcal{O}(|G|^2)$.
- За правила от вида $A \to u_1 u_2$, където $u_1, u_2 \in V \cup \Sigma$, заменяме всяка буква u_i с новата променлива U_i и добавяме правилото $U_i \to u_i$. Например, правилото $A \to aB$ се заменя с правилото $A \to XB$ и добавяме правилото $X \to a$, където X е нова променлива. Това можем да направим за време $\mathcal{O}(|G|)$ и новата граматика ще има дължина $\mathcal{O}(|G|)$.
- Ако искаме ε да бъде в езика на граматиката, то добавяме нова начална променлива S_0 и правило $S_0 \to_G \varepsilon$ и правилата $S_0 \to \alpha$ за $S \to \alpha$.

□

Теорема 3.6. При дадена безконтекстна граматика G, можем да намерим еквивалентна на нея граматика G' в нормална форма на Чомски за време $\mathcal{O}(|G|^2)$, като получената граматика е с дължина $\mathcal{O}(|G|^2)$.

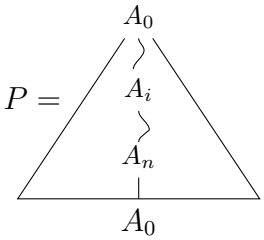
Теорема 3.7. Съществува *полиномиален* алгоритъм, който определя по дадена безконтекстна граматика G дали $\mathcal{L}(G)$ е безкраен език.

[HU79, стр. 137]. Важно е, че алгоритмът е полиномиален. ✍ Защо от *Лема 3.5* имаме експоненциален алгоритъм за проверка дали езикът на една граматика е безкраен?

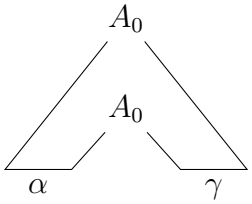
Доказателство. Нека е дадена една безконтекстна граматика G. Нека да разгледаме граматиката G' в НФЧ *без безполезни променливи*, за която $\mathcal{L}(G) = \mathcal{L}(G')$. От граматиката $G' = \langle V', \Sigma, S, R' \rangle$ строим граф с възли променливите от V' като за $A, B \in V'$ имаме ребро $A \to B$ в графа точно тогава, когато съществува $C \in V'$, за което $A \to_{G'} BC$ или $A \to_{G'} CB$ е правило в граматиката G. Сега ще съобразим, че ако в получения граф имаме цикъл, то $\mathcal{L}(G') = \infty$. Да разгледаме един такъв цикъл в графа:

$$A_0 \to A_1 \to A_2 \to \cdots \to A_n \to A_0.$$

Понеже граматиката е в нормална форма на Чомски, то съществуват думи $\lambda, \rho \in (V \cup \Sigma)^\star$, за които $A_0 \overset{n+1}{\lhd} \lambda A_0 \rho$ и $|\lambda\rho| = n + 1$



(а) $A_0 \overset{n+1}{\lhd} \lambda A_0 \rho$, където $\lambda, \rho \in (V \cup \Sigma)^\star$.



(б) $A_0 \overset{\geq n+1}{\lhd} \alpha A_0 \gamma$, където $\alpha, \gamma \in \Sigma^\star$.

Понеже в G няма безполезни променливи, то $\lambda \overset{\star}{\Rightarrow} \alpha$, където $\alpha \in \Sigma^\star$ и $|\lambda| \leq |\alpha|$ и $\rho \overset{\star}{\Rightarrow} \gamma$, където $\gamma \in \Sigma^\star$ и $|\rho| \leq |\gamma|$. Оттук заключаваме, че $A_0 \overset{\star}{\lhd} \alpha A_0 \gamma$ и $|\alpha\gamma| \geq 1$. Понеже A_0 не е безполезна променлива, то $A_0 \overset{\star}{\lhd} \beta$, за някоя дума $\beta \in \Sigma^\star$. Сега комбинираме *Твърдение 3.12* и *Твърдение 3.13* за да получим следния извод:

$$\frac{A_0 \overset{\star}{\lhd} \alpha A_0 \gamma \qquad A_0 \overset{\star}{\lhd} \beta \qquad i \in \mathbb{N}}{A_0 \overset{\star}{\lhd} \alpha^i \beta \gamma^i}$$

Понеже $|\alpha\beta| \geq 1$, заключаваме, че $\mathcal{L}(G)$ е безкраен език. Така доказахме, че ако в графът има цикли, то $\mathcal{L}(G)$ е безкраен език.

За обратната посока, нека в графът няма цикли. Да разгледаме една променлива A, от която най-дългият път има k на брой възли. От принципа на Дирихле знаем, че $k \leq |V|$. Това означава, че ако $A \overset{\star}{\lhd} \alpha$, за някоя $\alpha \in \Sigma^\star$, то $A \overset{\leq k}{\lhd} \alpha$ и понеже граматиката е в нормална форма на Чомски, то $|\alpha| \leq 2^{k-1}$. Оттук следва, че всички думи, които се извеждат от променливата A са най-много 2^{k-1} на брой. Заключаваме, че ако в графът няма цикли, то $\mathcal{L}(G)$ е краен. □

Теорема 3.8. Съществува *полиномиален* алгоритъм, който определя по дадена безконтекстна граматика G дали $\mathcal{L}(G)$ е краен език.

3.8.3 Проблемът за принадлежност

Теорема 3.9. Съществува *полиномиален* алгоритъм относно дължината на входната дума, който проверява дали дадена дума принадлежни на граматиката G.

Можем да приемем, че $G = \langle V, \Sigma, R, S\rangle$ е граматика в нормална форма на Чомски.

Това е алгоритъм на Cocke, Younger и Kasami (CYK), които откриват идеята за този алгоритъм независимо един от друг. Той е пример за динамично програмиране [Koz97, стр. 192], [Sha08, стр. 141]. При вход дума α, алгоритъмът работи за време $\mathcal{O}(|\alpha|^3)$.

Algorithm 2 Проверка дали $\alpha \in \mathcal{L}(G)$

```
1:  n = len(α)                                   ▷ Вход дума α = a_0 a_1 ⋯ a_{n-1}
2:  for all i < n do
3:      V[i][i] := {A ∈ V | A →_G α[i]}
4:      for all j := i + 1, ..., n − 1 do
5:          V[i][j] := ∅
6:  for all s := 1, ..., n − 1 do                ▷ Дължина на интервала
7:      for all i < n − s do                     ▷ Начало на интервала
8:          for all k := i, ..., i + s − 1 do    ▷ Разделяне на интервала
9:              if ∃(A, BC) ∈ R & B ∈ V[i][k] & C ∈ V[k+1][i+s] then
10:                 V[i][i+s] := V[i][i+s] ∪ {A}
11: if S ∈ V[0][n-1] then
12:     return True                              ▷ Има извод на думата от S
13: else
14:     return False
```

Лема 3.10. Нека е дадена безконтекстната граматика G в нормална форма на Чомски и думата α. Всеки път, точно преди да се изпълни ред (6) от програмата описана в Алгоритъм 2, е изпълнено, че за всяко $i < n - s$,

$$\texttt{V[i][i+s]} = \{A \in V \mid A \overset{\star}{\Rightarrow}_G a_i a_{i+1} \cdots a_{i+s}\}.$$

Доказателство. Пълна индукция по s. За $s = 0$ е ясно, защото от ред (3) и от факта, че граматиката е в нормална форма на Чомски следва, че за всяко $i < n$,

$$\texttt{V[i][i]} = \{A \in V \mid A \to_G \alpha[i]\} = \{A \in V \mid A \overset{\star}{\Rightarrow}_G a_i\}.$$

Сега ще докажем твърдението за $s > 0$, т.е. ще докажем, че за всяко $i < n - s$ е изпълнено, че:

$$\texttt{V[i][i+s]} = \{A \in V \mid A \overset{\star}{\Rightarrow}_G a_i a_{i+1} \cdots a_{i+s}\}.$$

Да разгледаме двете посоки на това равенство.

($\subseteq$) Нека $A \in$ `V[i][i+s]`. Единствената стъпка на алгоритъма, при която може да сме добавили променливата A към множеството `V[i][i+s]` е ред (10). Тогава имаме,

че съществува k, за което $i \leq k < i + s$, и

$$\begin{aligned} &B \in \texttt{V[i][k]}, && // \; k = i + \overbrace{(k-i)}^{s'} \\ &C \in \texttt{V[k+1][i+s]}, && // \; i + s = (k+1) + \underbrace{(i+s-k-1)}_{s''} \\ &A \to_G BC. \end{aligned}$$

Понеже $s' < s$ и $s'' < s$, от **(И.П.)** имаме, че

$$\begin{aligned} B &\stackrel{\star}{\Rightarrow}_G a_i a_{i+1} \cdots a_k \\ C &\stackrel{\star}{\Rightarrow}_G a_{k+1} \cdots a_{i+s}. \end{aligned}$$

Заключаваме веднага, че $A \stackrel{\star}{\Rightarrow}_G a_i a_{i+1} \cdots a_{i+s}$.

($\supseteq$) Нека $A \stackrel{\star}{\Rightarrow}_G a_i a_{i+1} \cdots a_{i+s}$ и да разгледаме първото правило, което сме приложили в този извод. Понеже G е в нормална форма на Чомски, правилото е от вида $A \to_G BC$ и тогава, според *Твърдение 3.6*, съществува k, за което $i \leq k < i + s$, и

$$\begin{aligned} B &\stackrel{\star}{\Rightarrow} a_i \cdots a_k \\ C &\stackrel{\star}{\Rightarrow} a_{k+1} \cdots a_{i+s}. \end{aligned}$$

От **(И.П.)** получаваме, че $B \in \texttt{V[i][k]}$ и $C \in \texttt{V[k+1][i+s]}$. Тогава от ред (10) на алгоритъма е ясно, че $A \in \texttt{V[i][i+s]}$.

□

Пример 3.13. Нека е дадена граматиката G в нормална форма на Чомски с правила

$$\begin{aligned} S &\to a \mid AB \mid AC \\ A &\to a \\ B &\to b \\ C &\to SB \mid AS. \end{aligned}$$

Ще приложим CYK алгоритъма за да проверим дали $aaabb \in \mathcal{L}(G)$.

- За $s = 0$ имаме, че:
  - $\texttt{V[0][0]} = \texttt{V[1][1]} = \texttt{V[2][2]} = \{S, A\}$;
  - $\texttt{V[3][3]} = \texttt{V[4][4]} = \{B\}$.
- За $s = 1$ имаме, че:
  - $\texttt{V[0][1]} = \texttt{V[1][2]} = \{C\}$;
  - $\texttt{V[2][3]} = \{S, C\}$;
  - $\texttt{V[3][4]} = \emptyset$.
- За $s = 2$ имаме, че:
  - $\texttt{V[0][2]} = \{S\} \cup \emptyset$;
  - $\texttt{V[1][3]} = \{S, C\} \cup \emptyset$;
  - $\texttt{V[2][4]} = \emptyset \cup \{C\}$.
- За $s = 3$ имаме, че:
  - $\texttt{V[0][3]} = \{S, C\} \cup \emptyset \cup \emptyset = \{S, C\}$;
  - $\texttt{V[1][4]} = \{S\} \cup \emptyset \cup \{C\} = \{S, C\}$.
- За $s = 4$ имаме, че:
  - $\texttt{V[0][4]} = \{S, C\} \cup \emptyset \cup \emptyset \cup \{C\} = \{S, C\}$.

Понеже $S \in \texttt{V[0][4]}$, то $aaabb \in \mathcal{L}(G)$.

3.9 Най-ляв извод в граматика

В нашата дефиниция на извод, изборът върху коя променлива да приложим правило от граматиката е недетерминистичен. В някои случаи, за нас ще бъде важно винаги да правим детерминистичен избор на това върху коя променлива прилагаме правило.

Най-левият извод ще бъде важен за нас, когато разгледаме стековите автомати.

Определение 3.2. За две думи $\alpha, \beta \in (V \cup \Sigma)^\star$, дефинираме **най-ляв извод** на думата β от думата α в граматиката, което ще записваме като $\alpha \Rightarrow_{\texttt{left}} \beta$, по следния начин:

$$\frac{(A, \alpha) \in R \qquad \lambda \in \Sigma^\star \qquad \rho \in (V \cup \Sigma)^\star}{\lambda A \rho \Rightarrow_{\texttt{left}} \lambda \alpha \rho}$$

Новото е, че имаме изискването $\lambda \in \Sigma^\star$. Това на практика означава, че на всяка стъпка заместваме възможно най-лявата променлива.

Интуитивно, $A \overset{\star}{\lhd} \alpha$ ни казва, че при обхождане на синтактичното дърво в широчина ние получаваме като листа думата α. Ако $A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha$, то обхождаме синтактичното дърво с корен A в дълбочина, като винаги избираме най-левия необходен клон. Ако $A \overset{\star}{\Rightarrow} \alpha$, то обхождаме синтактичното дърво в дълбочина без да имаме детерминистичен избор с кой необходен клон да продължим.

Твърдение 3.16. За произволно естествено число ℓ имаме извода:

$$\frac{\alpha \overset{\ell}{\Rightarrow}_{\texttt{left}} \beta \qquad \lambda \in \Sigma^\star \qquad \rho \in (V \cup \Sigma)^\star}{\lambda \alpha \rho \overset{\ell}{\Rightarrow}_{\texttt{left}} \lambda \beta \rho}$$

Твърдение 3.17. За произволни естествени числа ℓ_1 и ℓ_2 е изпълнено, че:

$$\frac{\alpha \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \beta \qquad \beta \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \gamma}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow}_{\texttt{left}} \gamma}$$

Твърдение 3.18. За произволни ℓ_1 и ℓ_2 имаме извода:

$$\frac{\alpha_1 \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \beta_1 \qquad \beta_1 \in \Sigma^\star \qquad \alpha_2 \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \beta_2}{\alpha_1\alpha_2 \overset{\ell_1+\ell_2}{\Rightarrow}_{\texttt{left}} \beta_1\beta_2}$$

Следваща *Лема 3.11* е важна, защото тя на практика ни казва, че няма значение в какъв ред ще заместваме променливите в един извод на безконтекстна граматика.

Лема 3.11. За всяка безконтекстна граматика G, променлива A и дума $\alpha \in \Sigma^\star$,

$$A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha \text{ точно тогава, когато } A \overset{\star}{\Rightarrow} \alpha.$$

В частност, $\mathcal{L}(G) = \{\alpha \in \Sigma^\star \mid S \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha\}$.

Доказателство. От правилата за извод е видно, че ако $A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha$, то $A \overset{\star}{\Rightarrow} \alpha$.

За другата посока, според *Теорема 3.2* е достатъчно да се докаже, че ако $A \overset{\star}{\lhd} \alpha$ то $A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha$. С пълна индукция по ℓ ще докажем, че за произволно число ℓ, ако $A \overset{\ell}{\lhd} \alpha$, то $A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha$. Да отбележим, че щом $A \in V$, а $\alpha \in \Sigma^\star$ е ясно, че $\ell > 0$. Това означава, че $A \overset{\ell}{\lhd} \alpha$, защото имаме:

$$\frac{A \to_G X_1X_2\cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \alpha_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \alpha_n}{A \overset{\ell}{\lhd} \underbrace{\alpha_1\cdots\alpha_n}_{\alpha},} \text{ (1)}$$

където $X_i \in V \cup \Sigma$ и $\ell = 1 + \sup\{\ell_1, \ldots, \ell_n\}$. Започваме така:

$$\text{(Твърдение 3.17)}\ \frac{A \to_G X_1X_2\cdots X_n \qquad \dfrac{\dfrac{X_1 \overset{\ell_1}{\lhd} \alpha_1}{X_1 \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha_1}\ \text{(И.П.)}}{X_1X_2\cdots X_n \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha_1X_2\cdots X_n}\ \text{(Твърдение 3.16)}}{A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha_1X_2\cdots X_n}$$

Продължаваме последователно за всяко $i < n$ да правим извода:

$$\text{(Твърдения 3.17 и 3.16)}\ \frac{A \overset{\star}{\Rightarrow}_{\texttt{left}} \overbrace{\alpha_1\cdots\alpha_{i-1}}^{\in\Sigma^\star} X_iX_{i+1}\cdots X_n \qquad \dfrac{X_i \overset{\ell_i}{\lhd} \alpha_i}{X_i \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha_i}\ \text{(И.П.)}}{A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha_1\cdots\alpha_{i-1}\alpha_iX_{i+1}\cdots X_n}$$

Завършваме със следния извод:

$$\text{(Твърдения 3.17 и 3.16)}\ \frac{A \overset{\star}{\Rightarrow}_{\texttt{left}} \overbrace{\alpha_1\alpha_2\cdots\alpha_{n-1}}^{\in\Sigma^\star} X_n \qquad \dfrac{X_n \overset{\ell_n}{\lhd} \alpha_n}{X_n \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha_n}\ \text{(И.П.)}}{A \overset{\star}{\Rightarrow}_{\texttt{left}} \underbrace{\alpha_1\alpha_2\cdots\alpha_{n-1}\alpha_n}_{\alpha}}$$

□

Най-десен извод

Единствената разлика между дефиницията на $\overset{\star}{\Rightarrow}_{\texttt{right}}$ и тази на $\overset{\star}{\Rightarrow}$ е, че тук изискваме $\rho \in \Sigma^\star$.

За две думи $\alpha, \beta \in (V \cup \Sigma)^\star$, дефинираме **най-десен извод** в граматиката G, $\alpha \Rightarrow_{\texttt{right}} \beta$, по следния начин:

$$\frac{(A, \alpha) \in R \qquad \lambda \in (V \cup \Sigma)^\star \qquad \rho \in \Sigma^\star}{\lambda A\rho \Rightarrow_{\texttt{right}} \lambda\alpha\rho}$$

Задача 3.19. Докажете, че за всяка безконтекстна граматика G, променлива A и дума $\alpha \in \Sigma^\star$, $A \overset{\star}{\Rightarrow}_{\texttt{right}} \alpha$ точно тогава, когато $A \overset{\star}{\Rightarrow} \alpha$.

3.10 Недетерминирани стекови автомати

Недетерминиран стеков автомат е седморка от вида

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle,$$

На англ. *Push-down automaton*. В този курс няма да разглеждаме детерминирани стекови автомати. Когато кажем стеков автомат, ще имаме предвид недетерминиран стеков автомат. Означаваме $\Sigma_\varepsilon \stackrel{\text{деф}}{=} \Sigma \cup \{\varepsilon\}$ и $\Gamma^{\leq 2} \stackrel{\text{деф}}{=} \{\varepsilon\} \cup \Gamma \cup \Gamma^2$.

където:

- Q е крайно множество от състояния;
- Σ е крайна входна азбука;
- Γ е крайна стекова азбука;
- $\sharp \in \Gamma$ е символ за дъно на стека;
- $\Delta : Q \times \Sigma_\varepsilon \times \Gamma \to \mathscr{P}(Q \times \Gamma^{\leq 2})$ е функция на преходите;
- $q_{\texttt{start}} \in Q$ е начално състояние;
- $q_{\texttt{accept}} \in Q$ е заключителното състояние.

Обикновено се взима Δ функцията да има сигнатура $\Delta : Q \times \Sigma_\varepsilon \times \Gamma \to \mathscr{P}(Q \times \Gamma^\star)$. Дефиницията на стеков автомат има много вариации, всички еквивалентни помежду си.

Конфигурация (или моментно описание) на изчислението със стеков автомат представлява тройка от вида $(q, \alpha, \gamma) \in Q \times \Sigma^\star \times \Gamma^\star$, т.е. автоматът се намира в състояние q, думата, която остава да се прочете е α, а съдържанието на стека е думата γ. Удобно е да въведем бинарната релация $\vdash_P$ над $Q \times \Sigma^\star \times \Gamma^\star$, която ще ни казва как моментното описание на автомата P се променя след изпълнение на една стъпка.

На англ. Instanteneous description

$$\frac{\Delta(q, b, A) \ni (p, \beta) \qquad b \in \Sigma}{(q, b\alpha, A\gamma) \vdash_P (p, \alpha, \beta\gamma)} \qquad \frac{\Delta(q, \varepsilon, A) \ni (p, \beta)}{(q, \alpha, A\gamma) \vdash_P (p, \alpha, \beta\gamma)}$$

Фигура 3.10: Правила за извършване на една стъпка в изчисление на стеков автомат

Сега можем дефинираме бинарната релация $\vdash_P^\ell$ над $Q \times \Sigma^\star \times \Gamma^\star$, която ни казва, че конфигурацията κ се променя до конфигурация κ' след изчисление от ℓ стъпки на стековия автомат:

$$\frac{}{\kappa \vdash_P^0 \kappa} \text{ (рефлексивност)} \qquad \frac{\kappa \vdash_P \kappa'' \qquad \kappa'' \vdash_P^\ell \kappa'}{\kappa \vdash_P^{\ell+1} \kappa'} \text{ (транзитивност)}$$

Дефинираме релацията $\vdash_P^\star$ като $\kappa \vdash_P^\star \kappa' \stackrel{\text{деф}}{\Leftrightarrow} (\exists \ell)[\kappa \vdash_P^\ell \kappa']$. Езикът $\mathcal{L}(P)$, който се разпознава от стековия автомат P дефинираме по следния начин:

Казано по-дискретно, $\vdash_P^\star$ е рефлексивното и транзитивно затваряне на релацията $\vdash_P$.

$$\mathcal{L}(P) \stackrel{\text{деф}}{=} \{\, \omega \in \Sigma^\star \mid (q_{\texttt{start}}, \omega, \sharp) \vdash_P^\star (q_{\texttt{accept}}, \varepsilon, \varepsilon) \,\}.$$

Възможно е да се дадат и други еквивалентни дефиниции на $\mathcal{L}(P)$.

Твърдение 3.19. За произволни ℓ_1 и ℓ_2 е изпълнено, че:

$$\frac{(p, \alpha_1, \gamma_1) \vdash^{\ell_1} (q, \varepsilon, \varepsilon) \qquad (q, \alpha_2, \gamma_2) \vdash^{\ell_2} (r, \varepsilon, \varepsilon)}{(p, \alpha_1\alpha_2, \gamma_1\gamma_2) \vdash^{\ell_1+\ell_2} (r, \varepsilon, \varepsilon)}$$

Твърдение 3.20. Ако $(q, \alpha, AB) \vdash^{\ell} (p, \varepsilon, \varepsilon)$, то съществуват ℓ_1, ℓ_2, състояние r и разбиване на α като $\alpha = \alpha_1\alpha_2$, така че:

$$(q, \alpha_1\alpha_2, AB) \vdash^{\ell_1} (r, \alpha_2, B) \vdash^{\ell_2} (p, \varepsilon, \varepsilon).$$

Примери

Пример 3.14. За езика $L = \{a^n b^n \mid n \in \mathbb{N}\}$, да разгледаме $P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}}\rangle$, където

- $Q \stackrel{\text{деф}}{=} \{q, p, f\}$;
- $q_{\texttt{start}} \stackrel{\text{деф}}{=} q$ и $q_{\texttt{accept}} \stackrel{\text{деф}}{=} f$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$ и $\Gamma \stackrel{\text{деф}}{=} \{\sharp, a\}$;
- Релацията на преходите Δ има следната дефиниция:

Тук получаваме детерминистичен стеков автомат.

(1) $\Delta(q, a, \sharp) \stackrel{\text{деф}}{=} \{(q, a\sharp)\}$;

(2) $\Delta(q, a, a) \stackrel{\text{деф}}{=} \{(q, aa)\}$; // трупаме a-та в стека

(3) $\Delta(q, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(f, \varepsilon)\}$; // трябва да разпознаем и думата ε

(4) $\Delta(q, b, a) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}$; // Започваме да четем само b-та

(5) $\Delta(p, b, a) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}$; // Чистим a-тата от стека

(6) $\Delta(p, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(f, \varepsilon)\}$.

(7) За всички останали тройки (r, x, y), нека $\Delta(r, x, y) \stackrel{\text{деф}}{=} \emptyset$.

Да видим как думата a^2b^2 се разпознава от стековия автомат P:

$$\begin{aligned}
(q, a^2b^2, \sharp) &\vdash_P (q, ab^2, a\sharp) && \text{// правило } (1)\\
&\vdash_P (q, b^2, aa\sharp) && \text{// правило } (2)\\
&\vdash_P (p, b, a\sharp) && \text{// правило } (4)\\
&\vdash_P (p, \varepsilon, \sharp) && \text{// правило } (5)\\
&\vdash_P (f, \varepsilon, \varepsilon) && \text{// правило } (6)
\end{aligned}$$

Получихме, че $(q_{\texttt{start}}, a^2b^2, \sharp) \vdash_P^\star (q_{\texttt{accept}}, \varepsilon, \varepsilon)$, откъдето следва, че $a^2b^2 \in \mathcal{L}(P)$.

а) Докажете с индукция по n, че за всяко естествено число n са изпълнени свойствата:

$$(q, a^n\beta, \sharp) \vdash_P^n (q, \beta, a^n\sharp) \tag{3.9}$$
$$(p, b^n, a^n\sharp) \vdash_P^n (p, \varepsilon, \sharp). \tag{3.10}$$

Заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете, че с индукция по n, че за всяко естествено число n са изпълнени свойствата:

$$(q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \gamma\sharp) \implies \alpha = \gamma = a^n \tag{3.11}$$

$$(p, \beta, \gamma\sharp) \vdash_P^n (p, \varepsilon, \sharp) \implies \beta = b^n \,\&\, \gamma = a^n. \tag{3.12}$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

Пример 3.15. Езикът $L = \{\, \omega\omega^{\mathtt{rev}} \mid \omega \in \{a,b\}^\star \,\}$ се разпознава от стеков автомат

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\mathtt{start}}, q_{\mathtt{accept}} \rangle,$$

където:

- $Q \stackrel{\text{деф}}{=} \{q, p, f\}$ и $q_{\mathtt{start}} \stackrel{\text{деф}}{=} q$, $q_{\mathtt{accept}} \stackrel{\text{деф}}{=} f$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$, $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$;
- Функцията на преходите Δ има следната дефиниция:

За всички липсващи твойки в дефиницията на Δ приемаме, че Δ връща $\emptyset$

  (1) $\Delta(q, x, \sharp) \stackrel{\text{деф}}{=} \{(q, x\sharp)\}$, където $x \in \{a, b\}$;

  (2) $\Delta(q, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(q, \varepsilon)\}$;

  (3) $\Delta(q, x, x) \stackrel{\text{деф}}{=} \{(q, xx), (p, \varepsilon)\}$, където $x \in \{a, b\}$;

  (4) $\Delta(q, a, b) \stackrel{\text{деф}}{=} \{(q, ab)\}$;

  (5) $\Delta(q, b, a) \stackrel{\text{деф}}{=} \{(q, ba)\}$;

  (6) $\Delta(p, x, x) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}$, където $x \in \{a, b\}$;

  (7) $\Delta(p, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(f, \varepsilon)\}$;

Основното наблюдение, което трябва да направим за да разберем конструкцията на автомата е, че всяка дума от вида $\omega\omega^{\mathtt{rev}}$ може да се запише като $\omega_1 aa\omega_1^{\mathtt{rev}}$ или $\omega_1 bb\omega_1^{\mathtt{rev}}$. Да видим защо P разпознава думата $abaaba$ с празен стек. Започваме по следния начин:

$$\begin{aligned}
(q, abaaba, \sharp) &\vdash_P (q, baaba, a\sharp) && \text{// правило } (1)\\
&\vdash_P (q, aaba, ba\sharp) && \text{// правило } (5)\\
&\vdash_P (q, aba, aba\sharp). && \text{// правило } (4)
\end{aligned}$$

Сега можем да направим два избора как да продължим. Състоянието p служи за маркер, което ни казва, че вече сме започнали да четем $\omega^{\mathtt{rev}}$. Поради тази причина, продължаваме така:

$$\begin{aligned}
(q, aba, aba\sharp) &\vdash_P (p, ba, ba\sharp) && \text{// правило } (3)\\
&\vdash_P (p, a, a\sharp) && \text{// правило } (6)\\
&\vdash_P (p, \varepsilon, \sharp) && \text{// правило } (6)\\
&\vdash_P (f, \varepsilon, \varepsilon). && \text{// правило } (7)
\end{aligned}$$

Да проиграем още един пример. Да видим защо думата aba не се извежда от автомата.

$$\begin{aligned}
(q, aba, \sharp) &\vdash_P (q, ba, a\sharp) && \text{// правило } (1)\\
&\vdash_P (q, a, ba\sharp) && \text{// правило } (5)\\
&\vdash_P (q, \varepsilon, aba\sharp). && \text{// правило } (4)
\end{aligned}$$

От последното моментно описание на автомата нямаме нито един преход, следователно думата aba не се разпознава от P.

✍ Докажете, че $L = \mathcal{L}(P)$!

За (3.13) приложете индукция по дължината на думата α. За индукционната стъпка разгледайте α като $\alpha = \alpha' x$.

а) Докажете с индукция по n, за всяко естествено число n са изпълнени свойствата:

$$|\alpha| = n \implies (q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \alpha^{\mathtt{rev}}\sharp) \tag{3.13}$$
$$|\beta| = n \implies (p, \beta, \beta\sharp) \vdash_P^n (p, \varepsilon, \sharp). \tag{3.14}$$

Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете с индукция по n, че за всяко естествено число n са изпълнени свойствата:

$$(q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \gamma\sharp) \implies \gamma = \alpha^{\mathtt{rev}} \,\&\, |\alpha| = n \tag{3.15}$$
$$(p, \beta, \gamma\sharp) \vdash_P^n (p, \varepsilon, \sharp) \implies \gamma = \beta \,\&\, |\beta| = n. \tag{3.16}$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

От *Задача 3.22* знаем, че този език е безконтекстен.

Пример 3.16. Езикът $L = \{ \omega \in \{a, b\}^\star \mid |\omega|_a = |\omega|_b \}$ се разпознава от стековия автомат $P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\mathtt{start}}, q_{\mathtt{accept}} \rangle$, където:

- $Q = \{q, f\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, b, \sharp\}$;
- $q_{\mathtt{start}} = q$ и $q_{\mathtt{accept}} = f$;
- Можем да дефинираме релацията на преходите Δ по следния начин:
  (1) $\Delta(q, \varepsilon, \sharp) = \{(f, \varepsilon)\}$;
  (2) $\Delta(q, x, \sharp) = \{(q, x\sharp)\}$, където $x \in \{a, b\}$;
  (3) $\Delta(q, x, x) = \{(q, xx)\}$, където $x \in \{a, b\}$;
  (4) $\Delta(q, a, b) = \{(q, \varepsilon)\}$;
  (5) $\Delta(q, b, a) = \{(q, \varepsilon)\}$.

Да видим защо думата $abbbaa \in \mathcal{L}(P)$.

$$\begin{aligned}
(q, abbbaa, \sharp) &\vdash_P (q, bbbaa,\ a\sharp) && \text{// правило (2)}\\
&\vdash_P (q, bbaa,\ \sharp) && \text{// правило (5)}\\
&\vdash_P (q, baa,\ b\sharp) && \text{// правило (2)}\\
&\vdash_P (q, aa, bb\sharp) && \text{// правило (3)}\\
&\vdash_P (q, a,\ b\sharp) && \text{// правило (4)}\\
&\vdash_P (q, \varepsilon,\ \sharp) && \text{// правило (4)}\\
&\vdash_P (f, \varepsilon,\ \varepsilon). && \text{// правило (1)}
\end{aligned}$$

а) Докажете с пълна индукция по дължината на думата γ, че за произволна дума $\gamma \in \{a, b\}^\star$, е изпълнено, че:

$$(\forall n)[a^n\gamma \in L \implies (q, \gamma, a^n\sharp) \vdash_P^\star (q, \varepsilon, \sharp)] \tag{3.17}$$
$$(\forall n)[b^n\gamma \in L \implies (q, \gamma, b^n\sharp) \vdash_P^\star (q, \varepsilon, \sharp)]. \tag{3.18}$$

Ще докажем едновременно *Свойство (3.17)* и *Свойство (3.18)* с пълна индукция по дължината на думата γ.

Да разгледаме произволна дума γ. Случаят, когато $|\gamma| = 0$ е тривиален. Нека $|\gamma| > 0$ и да приемем, че $a^n\gamma \in L$. Ясно е, че можем да представим γ като $\gamma = a^k b\gamma'$, за някое k.

- Ако $n+k=0$, то $a^n\gamma = b\gamma' \in L$ и прилагаме **(И.П.)** за *Свойство (3.18)* с думата γ' и получаваме, че:

$$\begin{aligned}(q, \overbrace{b\gamma'}^{\gamma}, \sharp) &\vdash (q, \gamma', b\sharp) && \text{// правило (2)}\\ &\vdash^\star (q, \varepsilon, \sharp) && \text{// \textbf{(И.П.)}}\end{aligned}$$

- Ако $n+k>0$, то $a^{n+k-1}\gamma' \in L$ и прилагаме **(И.П.)** за *Свойство (3.17)* с думата γ' и получаваме, че:

$$\begin{aligned}(q, \overbrace{a^k b\gamma'}^{\gamma}, a^n\sharp) &\vdash^\star_P (q, b\gamma', a^{n+k}\sharp) && \text{// правило (3)}\\ &\vdash^\star_P (q, \gamma', a^{n+k-1}\sharp) && \text{// правило (5)}\\ &\vdash^\star_P (q, \varepsilon, \sharp). && \text{// \textbf{(И.П.)}}\end{aligned}$$

Аналогично се доказва и *Свойство (3.18)*. Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете с пълна индукция по дължината на думата γ, че за произволна дума $\gamma \in \{a,b\}^\star$ е изпълнено, че:

$$(\forall n)[(q,\gamma,a^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp) \implies a^n\gamma \in L] \tag{3.19}$$
$$(\forall n)[(q,\gamma,b^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp) \implies b^n\gamma \in L]. \tag{3.20}$$

Нека имаме изчислението $(q,\gamma,a^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp)$. Да представим думата γ като $\gamma = a^k b\gamma'$.

- Ако $n+k=0$, то можем да разбием изчислението по следния начин:

$$\begin{aligned}(q, b\gamma', \sharp) &\vdash_P (q, \gamma', b\sharp)\\ &\vdash^\star (q, \varepsilon, \sharp).\end{aligned}$$

Тогава от **(И.П.)** за *Свойство (3.20)* следва, че $a^n\gamma = b\gamma' \in L$.

- Ако $n+k>0$, то можем да разбием изчислението по следния начин:

$$\begin{aligned}(q, a^k b\gamma', a^n\sharp) &\vdash^\star_P (q, b\gamma', a^{n+k}\sharp)\\ &\vdash_P (q, \gamma', a^{n+k-1}\sharp)\\ &\vdash^\star (q, \varepsilon, \sharp).\end{aligned}$$

Тогава от **(И.П.)** за *Свойство (3.19)* следва, че $a^{n+k-1}\gamma' \in L$, но оттук веднага получаваме, че $a^n a^k b\gamma' \in L$.

Аналогично се доказва и *Свойство (3.20)*. Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

Пример 3.17. Езикът $L = \{\, \omega \in \{a,b\}^\star \mid \omega \text{ е балансирана дума} \,\}$ се разпознава от стековия автомат $P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}}\rangle$, където:

От *Задача 3.23* знаем, че този език е безконтекстен.

- $Q = \{q, f\}$;
- $q_{\texttt{start}} = q$ и $q_{\texttt{accept}} = f$;
- $\Sigma = \{a,b\}$ и $\Gamma = \{a, \sharp\}$;
- Можем да дефинираме релацията на преходите Δ по следния начин:

✍ Докажете, че $L = \mathcal{L}(P)$!

(1) $\Delta(q,\varepsilon,\sharp) = \{(f,\varepsilon)\}$;

(2) $\Delta(q,a,\sharp) = \{(q,a\sharp)\}$;

(3) $\Delta(q,a,a) = \{(q,aa)\}$;

(4) $\Delta(q, b, a) = \{(q, \varepsilon)\}$;

Индукция по дължината на думата β.

(а) Докажете, че за произволно естествено число n и произволна дума $\beta \in \{a, b\}^\star$, е изпълнено, че:

$$a^n\beta \in L \implies (q, \beta, a^n\sharp) \vdash_P^\star (q, \varepsilon, \sharp).$$

Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

Индукция по броя на стъпките в изчислението на стековия автомат.

(б) Докажете, че за произволно естествено число n и произволна дума $\beta \in \{a, b\}^\star$, е изпълнено, че:

$$(q, \beta, a^n\sharp) \vdash_P^\star (q, \varepsilon, \sharp) \implies a^n\beta \in L.$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

3.11 Теорема за еквивалентност

Лема 3.12. За всяка безконтекстна граматика G, съществува стеков автомат P, такъв че $\mathcal{L}(G) = \mathcal{L}(P)$.

Доказателството на лемата следва до голяма степен [PL98, стр. 136].

Доказателство. Нека е дадена безконтекстната граматика $G = \langle V, \Sigma, R, S\rangle$ в нормална форма на Чомски. Нашата цел е да построим стеков автомат

Тук е важно да използваме най-ляв извод в граматика.

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}}\rangle,$$

който разпознава езика $\mathcal{L}(G)$.

- $Q = \{q_{\texttt{start}}, p, q_{\texttt{accept}}\}$;
- $\Gamma = \Sigma \cup V \cup \{\sharp\}$;
- Релацията на преходите Δ дефинираме по следния начин:

Понеже граматиката е в нормална форма на Чомски, то $|\alpha| \leq 2$ и удовлетворяваме дефиницията на Δ.

  (1) $\Delta(q_{\texttt{start}}, \varepsilon, \sharp) = \{(p, S\sharp)\}$;
  (2) $\Delta(p, \varepsilon, A) = \{(p, \gamma) \mid A \to_G \gamma\}$, за всяка променлива $A \in V$;
  (3) $\Delta(p, a, a) = \{(p, \varepsilon)\}$, за всяка буква $a \in \Sigma$;
  (4) $\Delta(p, \varepsilon, \sharp) = \{(q_{\texttt{accept}}, \varepsilon)\}$.

Ще докажем, че за всяка променлива $A \in V$, за всяка дума $\alpha \in \Sigma^\star$, и $\delta \in (V \cup \Sigma)^\star$, то е изпълнено, че:

(а) ако $A \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha$, то $(p, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon)$;

(б) ако $(p, \alpha, \delta) \vdash_P^\star (p, \varepsilon, \varepsilon)$, то $\delta \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha$.

Ако приемем, че (а) и (б) са изпълнени, тогава, ако вземем $\delta = S$ и $A = S$, то ще получим, че

$$\begin{aligned}
\alpha \in \mathcal{L}(G) &\Leftrightarrow S \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha \\
&\Leftrightarrow (p, \alpha, S) \vdash_P^\star (p, \varepsilon, \varepsilon) && \text{// от (а) и (б)} \\
&\Leftrightarrow (q_{\texttt{start}}, \alpha, \sharp) \vdash_P^\star (q_{\texttt{accept}}, \varepsilon, \varepsilon) && \text{// от деф. на } P \\
&\Leftrightarrow \alpha \in \mathcal{L}(P).
\end{aligned}$$

Сега преминаваме към доказателствата на двете твърдения.

Доказателството на (а) ще проведем с пълна индукция по дължината ℓ на извода $A \overset{\ell}{\Rightarrow}_{\texttt{left}} \alpha$ за $\ell \geq 1$.

Очевидно е, че не е възможно да имаме $A \overset{0}{\Rightarrow}_{\texttt{left}} \alpha$.

Нека $\ell = 1$. Този случай е лесен. Понеже граматиката е в нормална форма на Чомски, тук единствената възможност е да имаме извод $A \overset{1}{\Rightarrow}_{\texttt{left}} a$, за някоя $a \in \Sigma$. Тогава, според дефиницията на стековия автомат P, имаме следното изчисление:

$$(p, a, A) \vdash_P (p, a, a) \vdash_P (p, \varepsilon, \varepsilon).$$

Нека $\ell > 1$. Тогава изводът може да се разбие така:

$$\frac{\dfrac{A \to_G BC}{A \Rightarrow_{\texttt{left}} BC} \quad BC \overset{\ell-1}{\Rightarrow}_{\texttt{left}} \alpha}{A \overset{\ell}{\Rightarrow}_{\texttt{left}} \alpha}$$

Сега прилагаме *Твърдение 3.6* и *Лема 3.11* и получаваме, че можем да разбием извода $BC \overset{\ell-1}{\Rightarrow}_{\texttt{left}} \alpha$ така:

$$B \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \alpha_1$$
$$C \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \alpha_2,$$

където $\alpha = \alpha_1 \cdot \alpha_2$ и $\ell - 1 = \ell_1 + \ell_2$. Понеже $\ell_1 < \ell$ и $\ell_2 < \ell$, от индукционното предположение получаваме изчислението:

$$\frac{\dfrac{\dfrac{A \to_G BC}{\Delta(p, \varepsilon, A) \ni (p, BC)}}{(p, \alpha_1\alpha_2, A) \vdash_P (p, \alpha_1\alpha_2, BC)} \quad \textbf{(и.п.)} \quad \frac{\dfrac{B \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \alpha_1}{(p, \alpha_1, B) \vdash_P^\star (p, \varepsilon, \varepsilon)} \quad \dfrac{C \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \alpha_2}{(p, \alpha_2, C) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(p, \alpha_1\alpha_2, BC) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(p, \underbrace{\alpha_1\alpha_2}_{\alpha}, A) \vdash_P^\star (p, \varepsilon, \varepsilon)}$$

За (б), индукция по броя на стъпките ℓ в изчислението на стековия автомат.

Нека $\ell = 0$ и $(p, \alpha, \delta) \vdash_P^0 (p, \varepsilon, \varepsilon)$. Ясно е, че единствената възможност е $\alpha = \varepsilon$ и $\delta = \varepsilon$. Тогава $\varepsilon \overset{\star}{\Rightarrow}_{\texttt{left}} \varepsilon$.

Нека $\ell > 0$ и $(p, \alpha, \delta) \vdash_P^\ell (p, \varepsilon, \varepsilon)$. Имаме два варианта за първата стъпка в това изчисление.

Започваме със случая, който не зависи от правилата на граматиката. Това означава, че първата стъпка от изчислението $(p, \alpha, \delta) \vdash_P^\ell (p, \varepsilon, \varepsilon)$ е направена защото $\Delta(p, a, a) \ni (p, \varepsilon)$, за някоя буква a. Тогава със сигурност можем да представим думите α и δ като $\alpha = a\beta$ и $\delta = a\rho$, за някои β и ρ, и да разбием изчислението по следния начин:

$$\frac{\dfrac{\Delta(p, a, a) \ni (p, \varepsilon)}{(p, a\beta, a\rho) \vdash_P (p, \beta, \rho)} \quad (p, \beta, \rho) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(p, \underbrace{a\beta}_{\alpha}, \underbrace{a\rho}_{\delta}) \vdash_P^\ell (p, \varepsilon, \varepsilon)}$$

Сега можем да приложим индукционното предположение и да получим извода:

$$\dfrac{a \in \Sigma \qquad \dfrac{(p,\beta,\rho) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{\rho \overset{\star}{\Rightarrow}_{\texttt{left}} \beta}\ \textbf{(И.П.)}}{\underbrace{a\rho}_{\delta} \overset{\star}{\Rightarrow}_{\texttt{left}} \underbrace{a\beta}_{\alpha}}\ \textit{(Твърдение 3.16)}$$

Вторият случай зависи от правилата на граматиката. Това означава, че първата стъпка от изчислението $(p,\alpha,\delta) \vdash_P^{\ell} (p,\varepsilon,\varepsilon)$ е направена защото $\Delta(p,\varepsilon,A) \ni (p,\gamma)$. Според конструкцията на стековия автомат, това означава, че със сигурност имаме правилото $A \to_G \gamma$, а думата δ може да се представи като $\delta = A\rho$, за някое ρ, и изчислението може да се разбие по следния начин:

Понеже граматиката е в нормална форма на Чомски, то $1 \leq |\gamma| \leq 2$.

$$\dfrac{\dfrac{\Delta(p,\varepsilon,A) \ni (p,\gamma)}{(p,\alpha,A\rho) \vdash_P (p,\alpha,\gamma\rho)} \qquad (p,\alpha,\gamma\rho) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{(p,\alpha,\underbrace{A\rho}_{\delta}) \vdash_P^{\ell} (p,\varepsilon,\varepsilon)}$$

Сега прилагаме индукционното предположение и получаваме извода:

$$\dfrac{\dfrac{A \to_G \gamma}{A\rho \Rightarrow_{\texttt{left}} \gamma\rho} \qquad \dfrac{(p,\alpha,\gamma\rho) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{\gamma\rho \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha}\ \textbf{(И.П.)}}{\underbrace{A\rho}_{\delta} \overset{\star}{\Rightarrow}_{\texttt{left}} \alpha}$$

□

Пример 3.18. Нека е дадена безконтекстната граматика G с правила

$$\begin{aligned} S &\to AS \mid BS \mid \varepsilon \\ A &\to aA \mid a \\ B &\to Bb \mid b. \end{aligned}$$

Ще построим стеков автомат $P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle$, такъв че $\mathcal{L}(P) = \mathcal{L}(G)$.

- $\Sigma = \{a, b\}$;
- $\Gamma = \{A, S, B, a, b, \sharp\}$;
- $Q = \{q_{\texttt{start}}, q, q_{\texttt{accept}}\}$;
- Дефинираме релацията на преходите, следвайки конструкцията от *Теорема 3.10*:
  - $\Delta(q_{\texttt{start}}, \varepsilon, \sharp) = \{(q, S\sharp)\}$;
  - $\Delta(q, \varepsilon, S) = \{(q, AS), (q, BS), (q, \varepsilon)\}$;
  - $\Delta(q, \varepsilon, A) = \{(q, aA), (q, a)\}$;
  - $\Delta(q, \varepsilon, B) = \{(q, Bb), (q, b)\}$;
  - $\Delta(q, x, x) = \{(q, \varepsilon)\}$, където $x \in \{a, b\}$;
  - $\Delta(q, \varepsilon, \sharp) = \{(q_{\texttt{accept}}, \varepsilon)\}$;

Лема 3.13. За всеки стеков автомат P, съществува безконтекстна граматика G, такава че $\mathcal{L}(P) = \mathcal{L}(G)$.

Доказателство. Нека е даден стековия автомат

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle.$$

Тук основната идея е, че искаме променливата $[q, A, p]$ да кодира информацията, че когато стековият автомат е в състоянието p, и стекът съдържа само променливата A, то когато стековият автомат премине в състояние p стекът ще се изпразни. По-формално, съществува дума α, за която $(q, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon)$.

Ще дефинираме безконтекстна граматика G, за която $\mathcal{L}(P) = \mathcal{L}(G)$. Променливите на граматиката са

$$V = \{[q, A, p] \mid q, p \in Q \;\&\; A \in \Gamma\}.$$

Правилата на G са следните:

- Началната променлива е $S \stackrel{\text{деф}}{=} [q_{\texttt{start}}, \sharp, q_{\texttt{accept}}]$;
- Нека имаме $(r, BC) \in \Delta(q, a, A)$, където $a \in \Sigma_\varepsilon$. Тогава за всеки две състояния q' и p добавяме правилата:

$$[q, A, p] \to_G a[r, B, q'][q', C, p].$$

- Нека имаме $(r, B) \in \Delta(q, a, A)$, където $a \in \Sigma_\varepsilon$. Тогава за всяко състояние $p \in Q$ добавяме правилата:

$$[q, A, p] \to_G a[r, B, p].$$

- Нека имаме $(p, \varepsilon) \in \Delta(q, a, A)$, където $a \in \Sigma_\varepsilon$. Тогава добавяме правилата:

$$[q, A, p] \to_G a.$$

След като вече сме обяснили какви правила включва граматиката G, трябва да докажем, че за произволна дума $\alpha \in \Sigma^\star$, произволни състояния q и p, и произволен символ $A \in \Gamma$, е изпълнено, че:

$$[q, A, p] \stackrel{\star}{\Rightarrow}_{\texttt{left}} \alpha \text{ точно тогава, когато } (q, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon).$$

Да напомним, че $\Sigma_\varepsilon \stackrel{\text{деф}}{=} \Sigma \cup \{\varepsilon\}$.

($\Rightarrow$) С пълна индукция по броя на стъпките ℓ в изчислението на стековия автомат P ще докажем, че за произволно $\ell \geq 1$,

$$\text{ако } (q, \alpha, A) \vdash_P^\ell (p, \varepsilon, \varepsilon), \text{ то } [q, A, p] \stackrel{\star}{\Rightarrow}_{\texttt{left}} \alpha.$$

Ако $\ell = 1$, то е лесно, защото $\alpha = a \in \Sigma_\varepsilon$. Тогава $(p, \varepsilon) \in \Delta(q, a, A)$ и според конструкцията на граматиката G имаме правилото $[q, A, p] \to_G a$.

Ако $\ell > 1$, то в зависимост от първата стъпка на изчислението, имаме два случая. Нека $\alpha = a\beta$, където $a \in \Sigma_\varepsilon$.

- Ако $\Delta(q, a, A) \ni (r, B)$, то можем да разбием изчислението по следния начин:

$$\dfrac{\dfrac{\Delta(q,a,A) \ni (r,B)}{(q,a\beta,A) \vdash_P (r,\beta,B)} \quad (r,\beta,B) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{(q,\underbrace{a\beta}_{\alpha},A) \vdash_P^{\ell} (p,\varepsilon,\varepsilon)}$$

Сега можем да приложим индукционното предположение и да получим извода:

$$\dfrac{\dfrac{\dfrac{\Delta(q,a,A) \ni (r,B)}{[q,A,p] \to_G a[r,B,p]}\text{(деф.)}}{[q,A,p] \Rightarrow_{\texttt{left}} a[r,B,p]} \quad \dfrac{\dfrac{(r,\beta,B) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{[r,B,p] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta}\textbf{(и.п.)} \qquad a \in \Sigma}{a[r,B,p] \overset{\star}{\Rightarrow}_{\texttt{left}} a\beta}\textit{(Тв. 3.16)}}{[q,A,p] \overset{\star}{\Rightarrow}_{\texttt{left}} \underbrace{a\beta}_{\alpha}}$$

- Ако $\Delta(q,a,A) \ni (r,BC)$, то можем да разбием изчислението по следния начин:

$$\dfrac{\dfrac{\Delta(q,a,A) \ni (r,BC)}{(q,a\beta,A) \vdash_P (r,\beta,BC)} \quad (r,\beta,BC) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{(q,\underbrace{a\beta}_{\alpha},A) \vdash_P^{\ell} (p,\varepsilon,\varepsilon)}$$

Сега да видим как още можем разбием изчислението $(r,\beta,BC) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)$. Щом за $\ell-1$ стъпки достигаме от стек с големина 2 до празен стек, това означава, че можем да разбием думата β на две части като $\beta = \beta_1\beta_2$ със свойството, че след като прочетем β_1, то стекът има големина 1 и след като прочетем β_2, стекът вече е празен. Това означава, че съществува състояние q', за което можем да разбием изчислението на две части по следния начин:

Да обърнем внимание, че в междинните стъпки от двете изчисления, стекът може да расте.

$$\begin{aligned}&(r,\beta_1,B) \vdash_P^{\ell_1} (q',\varepsilon,\varepsilon)\\ &(q',\beta_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon),\\ &\ell_1+\ell_2 = \ell-1.\end{aligned}$$

Понеже $\ell_1 < \ell$ и $\ell_2 < \ell$, от индукционното предположение имаме следното:

$$\begin{aligned}(r,\beta_1,B) \vdash_P^{\ell_1} (q',\varepsilon,\varepsilon) &\implies [r,B,q'] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta_1\\ (q',\beta_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon) &\implies [q',C,p] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta_2.\end{aligned}$$

Понеже имаме, че $\Delta(q,a,A) \ni (r,BC)$, то според дефиницията на стековия автомат, в граматиката имаме правилото

$$[q,A,p] \to_G a[r,B,q'][q',C,p].$$

Обединявайки всичко, получаваме извода в граматиката:

$$\dfrac{\dfrac{\dfrac{\Delta(q,a,A) \ni (r,BC)}{[q,A,p] \to_G a[r,B,q'][q',C,p]}}{[q,A,p] \Rightarrow_{\texttt{left}} a[r,B,q'][q',C,p]} \quad \dfrac{\dfrac{(r,\beta_1,B) \vdash_P^{\ell_1} (q',\varepsilon,\varepsilon)}{[r,B,q'] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta_1}\ \textbf{(и.п.)}}{[r,B,q'] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta_1}\ \textit{(Тв. 3.18)} \quad \dfrac{(q',\beta_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon)}{[q',C,p] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta_2}\ \textbf{(и.п.)}}{[r,B,q'][q',C,p] \overset{\star}{\Rightarrow}_{\texttt{left}} \beta_1\beta_2}\ \textit{(Тв. 3.16)}}{a[r,B,q'][q',C,p] \overset{\star}{\Rightarrow}_{\texttt{left}} a\beta_1\beta_2}$$

$$[q,A,p] \overset{\star}{\Rightarrow}_{\texttt{left}} \underbrace{a\beta_1\beta_2}_{\alpha}$$

$(\Leftarrow)$ За тази посока, с пълна индукция по дължината на извода ℓ в граматиката G ще докажем, че за произволна дължина на извода $\ell \geq 1$,

$$\text{ако } [q,A,p] \overset{\ell}{\Rightarrow}_{\texttt{left}} \alpha, \text{ то } (q,\alpha,A) \vdash_P^{\star} (p,\varepsilon,\varepsilon).$$

- Нека $\ell = 1$. Тогава $\alpha = a \in \Sigma_\varepsilon$ и $[q,A,p] \to_G a$. Според дефиницията на граматиката, правилото $[q,A,p] \to_G a$ е добавено към граматиката, защото в стековият автомат имаме $\Delta(q,a,A) \ni (p,\varepsilon)$. Тогава е ясно, че $(q,a,A) \vdash_P (p,\varepsilon,\varepsilon)$.
- Нека $\ell > 1$. Тогава думата α може да се представи като $\alpha = a\beta$, където $a \in \Sigma_\varepsilon$, и според правилата на граматиката G имаме два случая.

  Тук отново е възможно $a = \varepsilon$. Това не е проблем, защото правим индукция по дължината на извода, а не по дължината на думата α.

  Първо, да приемем, че имаме следния извод:

$$\dfrac{\dfrac{[q,A,p] \to_G a[r,B,p]}{[q,A,p] \Rightarrow_{\texttt{left}} a[r,B,p]} \quad a[r,B,p] \overset{\ell-1}{\Rightarrow}_{\texttt{left}} a\beta}{[q,A,p] \overset{\ell}{\Rightarrow}_{\texttt{left}} \underbrace{a\beta}_{\alpha}}$$

  Сега можем да приложим индукционното предположение и да сглобим изчислението:

$$\dfrac{\dfrac{\dfrac{[q,A,p] \to_G a[r,B,p]}{\Delta(q,a,A) \ni (r,B)}\ \text{(деф. на } G)}{(q,a\beta,A) \vdash_P (r,\beta,B)} \quad \dfrac{\dfrac{a[r,B,p] \overset{\ell-1}{\Rightarrow}_{\texttt{left}} a\beta}{[r,B,p] \overset{\ell-1}{\Rightarrow}_{\texttt{left}} \beta}}{(r,\beta,B) \vdash_P^{\star} (p,\varepsilon,\varepsilon)}\ \textbf{(и.п.)}}{(q,\underbrace{a\beta}_{\alpha},A) \vdash_P^{\star} (p,\varepsilon,\varepsilon)}$$

  Сега да разгледаме втория случай:

$$\dfrac{\dfrac{[q,A,p] \to_G a[r,B,q'][q',C,p]}{[q,A,p] \Rightarrow_{\texttt{left}} a[r,B,q'][q',C,p]} \quad a[r,B,q'][q',C,p] \overset{\ell-1}{\Rightarrow}_{\texttt{left}} a\beta}{[q,A,p] \overset{\ell}{\Rightarrow}_{\texttt{left}} \underbrace{a\beta}_{\alpha}}$$

  Важно е, че $\beta \in \Sigma^\star$. Иначе няма да можем да приложим *Твърдение 3.6*.

Понеже $\beta \in \Sigma^\star$, можем да приложим *Твърдение 3.6* заедно с *Лема 3.11*, откъдето следва, че имаме разбиване на думата β като $\beta = \beta_1\beta_2$, където

$$\begin{aligned}&[r, B, q'] \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \beta_1\\ &[q', C, p] \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \beta_2,\\ &\ell_1 + \ell_2 = \ell - 1.\end{aligned}$$

Понеже $\ell_1 < \ell$ и $\ell_2 < \ell$, от индукционното предположение получаваме импликациите:

$$\begin{aligned}[r, B, q'] \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \beta_1 &\implies (r, \beta_1, B) \vdash_P^\star (q', \varepsilon, \varepsilon)\\ [q', C, p] \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \beta_2 &\implies (q', \beta_2, C) \vdash_P^\star (p, \varepsilon, \varepsilon).\end{aligned}$$

Правилото $[q, A, p] \to_G a[r, B, q'][q', C, p]$ е добавено в граматиката, защото $\Delta(q, a, A) \ni (r, BC)$. Обединявайки всичко, което знаем, получаваме изчислението:

$$\dfrac{\dfrac{\dfrac{[q, A, p] \to_G a[r, B, q'][q', C, p]}{\Delta(q, a, A) \ni (r, BC)}\ \text{(деф.)}}{(q, a\beta_1\beta_2, A) \vdash_P (r, \beta_1\beta_2, BC)} \quad \dfrac{\dfrac{[r, B, q'] \overset{\ell_1}{\Rightarrow}_{\texttt{left}} \beta_1}{(r, \beta_1, B) \vdash_P^\star (q', \varepsilon, \varepsilon)} \quad \dfrac{[q', C, p] \overset{\ell_2}{\Rightarrow}_{\texttt{left}} \beta_2}{(q', \beta_2, C) \vdash_P^\star (p, \varepsilon, \varepsilon)}\ \text{(И.П.)}}{(r, \beta_1\beta_2, BC) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(q, \underbrace{a\beta_1\beta_2}_{\alpha}, A) \vdash_P^\star (p, \varepsilon, \varepsilon)}\ \text{(транз.)}$$

□

Пример 3.19. Да разгледаме стековия автомат от Пример 3.17.

- Началната променлива в граматиката е $S = [q, \sharp, f]$.
- Понеже $\Delta(q, \varepsilon, \sharp) = \{(f, \varepsilon)\}$, то имаме правилото $[q, \sharp, f] \to_G \varepsilon$.
- Понеже $\Delta(q, a, \sharp) = \{(q, a\sharp)\}$, то имаме правилата $[q, \sharp, p] \to_G a[q, a, r][r, \sharp, p]$, за произволни състояния $r, p \in \{q, f\}$.
- Понеже $\Delta(q, a, a) = \{(q, aa)\}$, то имаме правилата $[q, a, p] \to_G a[q, a, r][r, a, p]$, за произволни състояния $r, p \in \{q, f\}$.
- Понеже $\Delta(q, b, a) = \{(q, \varepsilon)\}$, то имаме правилото $[q, a, q] \to_G b$.

Предишните две леми ни дават следната еквивалентност.

Теорема 3.10. Класът на езиците, които се разпознават от недетерминиран стеков автомат съвпада с класа на безконтекстните езици.

Сечение с регулярен език

От *Твърдение 3.15* знаем, че безконтекстните езици не са затворени относно операцията сечение, т.е. възможно е L_1 и L_2 да са безконтекстни езици, но $L_1 \cap L_2$ да не е безконтекстен. Оказва се обаче, че безконтекстните езици са затворени относно сечение с регулярен език.

Теорема 3.11. Нека L е безконтекстен език и R е регулярен език. Тогава тяхното сечение $L \cap R$ е безконтекстен език.

Тук адаптираме доказателството от [PL98, стр. 144].

Упътване. Нека имаме стеков автомат

$$P = \langle Q', \Sigma, \Gamma, \sharp, \Delta', q'_{\texttt{start}}, q'_{\texttt{accept}} \rangle, \text{ където } \mathcal{L}(P) = L,$$

и детерминиран краен автомат

$$\mathcal{A} = \langle Q'', \Sigma, q''_{\texttt{start}}, \delta'', F'' \rangle, \text{ където } \mathcal{L}(\mathcal{A}) = R.$$

Сравнете с конструкцията от *Твърдение 2.2*.

Ще определим нов стеков автомат $\mathcal{M} = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle$, където:

- $Q \overset{\text{деф}}{=} Q' \times Q''$;
- $q_{\texttt{start}} \overset{\text{деф}}{=} \langle q'_{\texttt{start}}, q''_{\texttt{start}} \rangle$;
- $F \overset{\text{деф}}{=} \{q'_{\texttt{accept}}\} \times F''$;
- Функцията на преходите Δ е дефинирана както следва:
  - Ако $(r_1, \gamma) \in \Delta'(q_1, a, Y)$, то
    $$\Delta(\langle q_1, q_2 \rangle, a, Y) \ni (\langle r_1, \delta''(q_2, a) \rangle, \gamma).$$
    Симулираме едновременно изчислението и на двата автомата.
  - Ако $(r_1, \gamma) \in \Delta'(q_1, \varepsilon, Y)$ и всяко $q_2 \in Q''$, то
    $$\Delta(\langle q_1, q_2 \rangle, \varepsilon, Y) \ni (\langle r_1, q_2 \rangle, \gamma).$$
    Нищо не четем от входната дума, следователно правим празен ход на $\mathcal{A}$
  - Δ не съдържа други преходи;
- Докажете, че ако $(\langle q_1, q_2 \rangle, \alpha, \gamma) \vdash^\star_{\mathcal{M}} (\langle p_1, p_2 \rangle, \varepsilon, \varepsilon)$, то
  $$(q_1, \alpha, \gamma) \vdash^\star_P (p_1, \varepsilon, \varepsilon) \text{ и } (q_2, \alpha) \vdash^\star_{\mathcal{A}} (p_2, \varepsilon).$$
- Докажете, че ако $(q_1, \alpha, \gamma) \vdash^\star_P (p_1, \varepsilon, \varepsilon)$ и $(q_2, \alpha) \vdash^\star_{\mathcal{A}} (p_2, \varepsilon)$, то
  $$(\langle q_1, q_2 \rangle, \alpha, \gamma) \vdash^\star_{\mathcal{M}} (\langle p_1, p_2 \rangle, \varepsilon, \varepsilon).$$

✍ Докажете, че $\mathcal{L}(\mathcal{M}) = \mathcal{L}(P) \cap \mathcal{L}(\mathcal{A})$!

□

Теорема 3.11 е удобна, когато искаме да докажем, че даден език не е безконтекстен. С нейна помощ можем да сведем езика до друг, за който вече знаем, че не е безконтекстен.

Пример 3.20. Да разгледаме езика $L = \{\omega \in \{a, b, c\}^\star \mid |\omega|_a = |\omega|_b = |\omega|_c\}$. Да допуснем, че L е безконтекстен език. Тогава, според *Теорема 3.11*, $L' = L \cap \mathcal{L}(a^\star b^\star c^\star)$ също е безконтекстен език. Но $L' = \{a^n b^n c^n \mid n \in \mathbb{N}\}$, за който знаем от *Пример 3.9*, че *не* е безконтекстен. Достигнахме до противоречие. Следователно, L не е безконтекстен език.

3.12 Допълнителни задачи

3.12.1 Равен брой леви и десни скоби

Нека за по-голяма яснота да положим

За да се приближим малко до по-реален пример, можете да си мислите, че тук искаме да разпознаем думите с равен брой леви и десни скоби, като например интерпретираме a като символа `{` и b като символа `}`.

$$\texttt{left}(\alpha) \stackrel{\text{деф}}{=} |\alpha|_a \qquad \text{// брой срещания на } a \text{ в } \alpha$$
$$\texttt{right}(\alpha) \stackrel{\text{деф}}{=} |\alpha|_b \qquad \text{// брой срещания на } b \text{ в } \alpha$$

Задача 3.20. Нека ω е произволна дума над азбуката $\{a, b\}$. Тогава:

а) ако $\texttt{left}(\omega) = \texttt{right}(\omega) + 1$, то съществуват думи ω_1, ω_2, за които е изпълнено:

- $\omega = \omega_1 a \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

б) ако $\texttt{right}(\omega) = \texttt{left}(\omega) + 1$, то съществуват думи ω_1, ω_2, за които е изпълнено:

- $\omega = \omega_1 b \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

Упътване. Ще се съсредоточим върху случая, когато ω е дума, за която $\texttt{left}(\omega) = \texttt{right}(\omega) + 1$. Ще докажем а) с индукция по дължината на думата.

Другият случай е аналогичен

- $|\omega| = 1$. Тогава $\omega_1 = \omega_2 = \varepsilon$ и $\omega = a$.
- Да приемем, че твърдението а) е вярно за думи с дължина $\leq n$.
- $|\omega| = n + 1$. Ще разгледаме два случая, в зависимост от първия символ на ω.
  - Случаят $\omega = a\omega'$ е очевиден. (Защо?)
  - Интересният случай е $\omega = b^i b a \omega'$, за някое i. Да разгледаме думата ω'', която се получава от ω като премахнем първото срещане на думата ba, т.е. $\omega'' = b^i \omega'$ и $|\omega''| = n - 1$. Понеже от ω сме премахнали равен брой леви и десни скоби, то $\texttt{left}(\omega'') = \texttt{right}(\omega'') + 1$. Според **(И.П.)** за ω'' са изпълнени свойствата:
    * $\omega'' = \omega_1'' a \omega_2''$;
    * $\texttt{left}(\omega_1'') = \texttt{right}(\omega_1'')$;
    * $\texttt{left}(\omega_2'') = \texttt{right}(\omega_2'')$.

    Понеже b^i е префикс на ω_1'', за да получим обратно ω, трябва да прибавим премахнатата част ba веднага след b^i в ω_1''.

□

Задача 3.21. За произволна дума $\omega \in \{a,b\}^\star$, докажете, че ако $\texttt{left}(\omega) > \texttt{right}(\omega)$, то съществуват думи ω_1 и ω_2, за които са изпълнени свойствата:

- $\omega = \omega_1 a \omega_2$;
- $\texttt{left}(\omega_1) \geq \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) \geq \texttt{right}(\omega_2)$.

Задача 3.22. Да се докаже, че езикът $L = \{\, \alpha \in \{a,b\}^\star \mid \texttt{left}(\alpha) = \texttt{right}(\alpha) \,\}$ е безконтекстен.

Алтернативна граматика за езика L е

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

Доказателство. Една възможна граматика G е следната:

$$S \to aSbS \mid bSaS \mid \varepsilon.$$

Като следствие от *Задача 3.20* може лесно да се изведе, че за думи ω, за които $\texttt{left}(\omega) = \texttt{right}(\omega)$, е изпълнено следното:

а) ако $\omega = a\omega'$, то са изпълнени свойствата:

- $\omega = a\omega_1 b \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

б) ако $\omega = b\omega'$, то са изпълнени свойствата:

- $\omega = b\omega_1 a \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

Да напомним, че $\mathcal{L}^\ell_G(S) = \{\omega \in \Sigma^\star \mid S \overset{\leq \ell}{\lhd} \omega\}$ и според *Твърдение 3.11* имаме следната рекурсивна връзка:

$$\begin{aligned}
\mathcal{L}^0_G(S) =& \emptyset \\
\mathcal{L}^{\ell+1}_G(S) =& \{a\} \cdot \mathcal{L}^\ell_G(S) \cdot \{b\} \cdot \mathcal{L}^\ell_G(S) \cup \\
& \{b\} \cdot \mathcal{L}^\ell_G(S) \cdot \{a\} \cdot \mathcal{L}^\ell_G(S) \cup \\
& \{\varepsilon\}.
\end{aligned}$$

За коректност на граматиката трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}^\ell_G(S) \subseteq \{\omega \in \{a,b\}^\star \mid \texttt{left}(\omega) = \texttt{right}(\omega)\}. \tag{3.21}$$

Очевидно е, че *Свойство (3.21)* е изпълнено за $\ell = 0$. Да приемем, че *Свойство (3.21)* е изпълнено за някое ℓ. Ще докажем *Свойство (3.21)* за $\ell + 1$. Да вземем произволна дума $\omega \in \mathcal{L}^{\ell+1}_G(S)$. Имаме три случая.

- Ако $\omega = \varepsilon$. Тогава е ясно, че $\texttt{left}(\omega) = \texttt{right}(\omega)$.
- Ако $\omega \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{b\} \cdot \mathcal{L}_G^\ell(S)$. Тогава $\omega = a\omega_1 b\omega_2$ и $\omega_1, \omega_2 \in \mathcal{L}_G^\ell(S)$. От **(И.П.)** получаваме, че $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$ и $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$. Заключаваме, че $\texttt{left}(\omega) = \texttt{right}(\omega)$
- Случаят, когато $\omega = b\omega_1 a\omega_2$, е аналогичен.

Така доказахме коректност на граматиката, т.е. имаме следното свойство:

$$\mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) \subseteq \{\omega \in \{a,b\}^\star \mid \texttt{left}(\omega) = \texttt{right}(\omega)\}.$$

Сега за пълнота на граматиката трябва да докажем, че

$$\{\omega \in \{a,b\}^\star \mid \texttt{left}(\omega) = \texttt{right}(\omega)\} \subseteq \mathcal{L}_G(S). \tag{3.22}$$

Това ще направим с пълна индукция по дължината на думите. Да вземем произволна дума $\omega \in \{a,b\}^\star$ и $\texttt{left}(\omega) = \texttt{right}(\omega)$. Да видим защо $\omega \in \mathcal{L}_G(S)$.

- Ако $\omega = \varepsilon$, то е ясно, че $\omega \in \mathcal{L}_G(S)$.
- Нека $\omega \neq \varepsilon$. Според *Задача 3.20* имаме два случая.
  - Нека $\omega = a\omega_1 b\omega_2$, $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$ и $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$. Тогава от **(И.П.)** имаме, че $\omega_1 \in \mathcal{L}_G(S)$ и $\omega_2 \in \mathcal{L}_G(S)$. Заключаваме, че
  $$\omega \in \{a\} \cdot \mathcal{L}_G(S) \cdot \{b\} \cdot \mathcal{L}_G(S) \subseteq \mathcal{L}_G(S).$$
  - Ако $\omega = b\omega_1 a\omega_2$, то с аналогични разсъждения получаваме, че
  $$\omega \in \{b\} \cdot \mathcal{L}_G(S) \cdot \{a\} \cdot \mathcal{L}_G(S) \subseteq \mathcal{L}_G(S).$$

Така доказахме пълнота на граматиката, т.е. имаме следното свойство:

$$\{\omega \in \{a,b\}^\star \mid \texttt{left}(\omega) = \texttt{right}(\omega)\} \subseteq \mathcal{L}_G(S).$$

□

3.12.2 Балансирани скоби

Нека α е дума над азбука, която включва буквите a и b. Ще казваме, че α е **балансирана**, ако са изпълнени свойствата:

- $\texttt{left}(\alpha) = \texttt{right}(\alpha)$;
- За всеки префикс γ на α, $\texttt{left}(\gamma) \geq \texttt{right}(\gamma)$.

Например, думата $aabaabbb$ е балансирана, докато $abbaaab$ не е балансирана. Практическият смисъл на тази задача е, че можем да напишем граматика, която да разпознава дали за всяка отваряща скоба {, която прочетем, по-късно ще прочетем и съответната затваряща скоба }.

Задача 3.23. Докажете, че $L = \{ \alpha \in \{a,b\}^\star \mid \alpha$ е балансирана дума $\}$ е безконтекстен език.

[Koz97, стр. 135]
✍ Докажете, че езикът L не е регулярен! Алтернативна граматика е

$S \to aSbS \mid \varepsilon.$

Доказателство. Да разгледаме граматиката G с правила

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

Ще докажем, че $L = \mathcal{L}(G)$. Имаме, че

$$\begin{aligned} \mathcal{L}_G^0(S) =& \emptyset \\ \mathcal{L}_G^{\ell+1}(S) =& \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{b\} \cup \\ & \mathcal{L}_G^\ell(S) \cdot \mathcal{L}_G^\ell(S) \cup \\ & \{\varepsilon\}. \end{aligned}$$

Първо да разгледаме коректност на граматиката, т.е. трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}_G^\ell(S) \subseteq \{ \alpha \in \{a,b\}^\star \mid \alpha \text{ е балансирана дума } \}. \tag{3.23}$$

Твърдението е очевидно за $\ell = 0$. Да приемем, че *Свойство (3.23)* е изпълнено за някое ℓ. Ще докажем *Свойство (3.23)* за $\ell + 1$. Да разгледаме произволна дума $\omega \in \mathcal{L}_G^{\ell+1}(S)$. Имаме три случая.

- Нека $\omega = \varepsilon$. Тогава е ясно, че ω е балансирана дума.
- Нека $\omega \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{b\}$. Тогава $\omega = a\omega_1 b$ и $\omega_1 \in \mathcal{L}_G^\ell(S)$. От **(И.П.)** имаме, че ω_1 е балансирана. Лесно се съобразява, че ω също е балансирана.
- Нека $\omega \in \mathcal{L}_G^\ell(S) \cdot \mathcal{L}_G^\ell(S)$. Тогава $\omega = \omega_1\omega_2$, такива че $\omega_1, \omega_2 \in \mathcal{L}_G^\ell(S)$. От **(И.П.)** имаме, че ω_1 и ω_2 са балансирани. Лесно се съобразява, че ω също е балансирана.

Така доказахме, че

$$\mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) \subseteq \{ \alpha \in \{a,b\}^\star \mid \alpha \text{ е балансирана дума } \}.$$

Сега ще докажем пълнота на граматиката, т.е. следното свойство:

$$\{ \alpha \in \{a,b\}^\star \mid \alpha \text{ е балансирана дума } \} \subseteq \mathcal{L}_G(S). \tag{3.24}$$

Това ще направим с *пълна* индукция по дължината на думите. Да разгледаме произволна балансирана дума ω. Имаме няколко случая, които трябва да разгледаме.

- Ако $\omega = \varepsilon$, то е ясно, че $\omega \in \mathcal{L}_G(S)$.

- Нека сега $\omega \neq \varepsilon$. Ясно е, че със сигурност $\omega = a\omega_1 b$. Проблемът е, че в общия случай не е ясно дали можем да приложим **(И.П.)** за ω_1, защото е възможно ω_1 да не е балансирана. Например, ако $\omega = abab$, то $\omega_1 = ba$ не е балансирана. Поради тази причина, трябва да сме по-внимателни и да разгледаме два допълнителни случая.

  – Нека ω има *същински* префикс ω_1, който да е балансирана дума. Понеже ω е балансирана дума, лесно се съобразява, че $\omega = \omega_1\omega_2$, ω_2 също е балансирана дума. Сега можем да приложим **(И.П.)** за ω_1 и ω_2 и да получим, че $\omega_1 \in \mathcal{L}_G(S)$ и $\omega_2 \in \mathcal{L}_G(S)$. Ясно е, че $\omega \in \mathcal{L}_G(S)$.

  Тук $\omega_1 \neq \varepsilon$ и $\omega_1 \neq \alpha$. Например, ab е същински префикс на $abab$, който е балансирана дума.

  – Нека ω да няма същински префикс ω_1, който е балансирана дума. Ясно е, че тогава $\omega = a\beta b$, за някое β. Да видим защо β е балансирана дума. Ако β е балансирана, то ще можем да приложим **(И.П.)** за β и ще сме готови.

  Например, $aabb$ няма същински префикс, който да е балансирана дума.

  За всеки префикс γ на β имаме, че $a\gamma$ е префикс на α, и понеже α е балансирана, то $\texttt{left}(a\gamma) \geq \texttt{right}(a\gamma)$. Възможно ли е $\texttt{left}(\gamma) < \texttt{right}(\gamma)$? Това може да се случи единствено ако $\texttt{left}(a\gamma) = \texttt{right}(a\gamma)$. Но тогава $a\gamma$ е същински префикс на α, за който $a\gamma$ е балансирана дума, което противоречи на случая, който разглеждаме. Това означава, че за произволен префикс γ на β, $\texttt{left}(\gamma) \geq \texttt{right}(\gamma)$ и оттук β е балансирана дума и можем да приложим **(И.П.)** . Тогава $\beta \in \mathcal{L}_G(S)$ и следователно $\alpha \in \{a\} \cdot \mathcal{L}_G(S) \cdot \{b\} \subseteq \mathcal{L}_G(S)$.

Така доказахме *Свойство (3.24)*, което ни дава пълнота на граматиката спрямо езика. □

3.12.3 Лесни задачи

Задача 3.24. Постройте регулярен израз за езика на следната граматика:

$$\begin{aligned} S &\to S + S \mid S * S \mid A \\ A &\to KL \mid LK \\ K &\to 0K \mid \varepsilon \\ L &\to 1K \mid \varepsilon. \end{aligned}$$

Задача 3.25. Докажете, че следните езици са безконтекстни.

а) $L = \{\omega \in \{a,b\}^\star \mid \omega = \omega^{\texttt{rev}}\}$;

б) $L = \{a^n b^{2m} c^n \mid m, n \in \mathbb{N}\}$;

в) $L = \{a^n b^m c^m d^n \mid m, n \in \mathbb{N}\}$;

г) $L = \{a^n b^{2k} \mid n, k \in \mathbb{N} \,\&\, n \neq k\}$;

д) $L = \{a^n b^k \mid n > k\}$;

е) $L = \{a^n b^k \mid n \geq 2k\}$;

ж) $L = \{a^n b^k c^m \mid n + k \geq m + 1\}$;

з) $L = \{a^n b^k c^m \mid n + k \geq m + 2\}$;

и) $L = \{a^n b^k c^m \mid n + k + 1 \geq m\}$;

а) $S \to aSa \mid bSb \mid a \mid b \mid \varepsilon$;

г) Обединение на два езика;

д) $S \to aSb \mid aS \mid a$;

ж) $S \to aSc \mid aS \mid aB \mid bB$,
$B \to bBc \mid bB \mid \varepsilon$;

и) $S \to aSc \mid aS \mid B \mid Bc$,
$B \to bBc \mid bB \mid \varepsilon$;

н) Обединение на три езика;

п) $S \to EaE$,
$E \to aEbE \mid bEaE \mid \varepsilon$;

к) $L = \{a^n b^m c^{2k} \mid n \neq 2m \,\&\, k \geq 1\}$;

л) $L = \{a^n b^k c^m \mid n + k \leq m\}$;

м) $L = \{a^n b^k c^m \mid n + k \leq m + 1\}$;

н) $L = \{a^n b^m c^k \mid n, m, k$ не са страни на триъгълник$\}$;

о) $L = \{a, b\}^\star \setminus \{a^{2n} b^n \mid n \in \mathbb{N}\}$;

п) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a = |\omega|_b + 1\}$;

р) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a \geq |\omega|_b\}$;

с) $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a > |\omega|_b\}$;

т) $L = \{\alpha \in \{a, b\}^\star \mid$ във всеки префикс β на α, $|\beta|_b \leq |\beta|_a\}$;

у) $L = \{\alpha\sharp\beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, \alpha^{\mathtt{rev}}$ е поддума на $\beta\}$.

ф) $L = \{\omega_1\sharp\omega_2\sharp\cdots\sharp\omega_n \mid n \geq 2 \,\&\, \omega_1, \omega_2, \ldots, \omega_n \in \{a, b\}^\star \,\&\, |\omega_1| = |\omega_2|\}$;

х) $L = \{\omega_1\sharp\omega_2\sharp\cdots\sharp\omega_n \mid n \geq 2 \,\&\, \omega_1, \ldots, \omega_n \in \{a, b\}^\star \,\&\, (\exists i \neq j)[\,|\omega_i| = |\omega_j|\,]\}$;

ц) $L = \{\omega_1\sharp\omega_2\sharp\cdots\sharp\omega_n \mid n \geq 2 \,\&\, (\forall i \in [1, n])[\,\omega_i \in \{a, b\}^\star \,\&\, |\omega_i| = |\omega_{n+1-i}|\,]\}$.

Задача 3.26. Проверете дали следните езици са безконтекстни:

а) $\{a^n b^{2n} c^{3n} \mid n \in \mathbb{N}\}$;

б) $\{a^n b^k c^k a^n \mid k \leq n\}$;

в) $\{a^n b^m c^k \mid n < m < k\}$;

г) $\{a^n b^n c^k \mid n \leq k \leq 2n\}$;

д) $\{a^n b^m c^k \mid k = \min\{n, m\}\}$;

е) $\{a^n b^n c^m \mid m \leq n\}$;

ж) $\{a^n b^m c^k \mid k = n \cdot m\}$;

з) $L^\star$, където $L = \{\alpha\alpha^{\mathtt{rev}} \mid \alpha \in \{a, b\}^\star\}$;

и) $\{\omega\omega\omega \mid \omega \in \{a, b\}^\star\}$;

к) $\{a^{n^2} b^n \mid n \in \mathbb{N}\}$;

л) $\{a^p \mid p$ е просто $\}$;

м) $\{\omega \in \{a, b\}^\star \mid \omega = \omega^{\mathtt{rev}}\}$;

н) $\{\omega^n \mid \omega \in \{a, b\}^\star \,\&\, |\omega|_b = 2 \,\&\, n \in \mathbb{N}\}$;

о) $\{\omega c^n \omega^{\mathtt{rev}} \mid \omega \in \{a, b\}^\star \,\&\, n = |\omega|\}$;

п) $\{\alpha c \beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, \alpha$ е подниз на $\beta\}$;

р) $\{\omega_1\sharp\omega_2\sharp\ldots\sharp\omega_k \mid k \geq 2 \,\&\, \omega_i \in \{a\}^\star \,\&\, (\exists i, j)[i \neq j \,\&\, \omega_i = \omega_j]\}$;

с) $\{\omega_1\sharp\omega_2\sharp\ldots\sharp\omega_k \mid k \geq 2 \,\&\, \omega_i \in \{a\}^\star \,\&\, (\forall i, j \leq k)[i \neq j \;\Leftrightarrow\; \omega_i \neq \omega_j]\}$;

т) $\{a^i b^j c^k \mid i, j, k \geq 0 \,\&\, (i = j \vee j = k)\}$;

у) $\{\omega \in \{a, b, c\}^\star \mid |\omega|_a > |\omega|_b > |\omega|_c\}$;

ф) $\{a, b\}^\star \setminus \{a^n b^n \mid n \in \mathbb{N}\}$;

х) $\{a^n b^m c^k \mid m^2 = 2nk\}$;

ц) $L = \{a^n b^m c^m a^n \mid m, n \in \mathbb{N} \,\&\, n = m + 42\}$;

ч) $L = \{\sharp a\sharp aa\sharp aaa\sharp\cdots\sharp a^{n-1}\sharp a^n\sharp \mid n \geq 1\}$;

ш) $\{a^m b^n c^k \mid m = n \vee n = k \vee m = k\}$;

щ) $\{a^m b^n c^k \mid m \neq n \vee n \neq k \vee m \neq k\}$;

ю) $\{\omega \in \{a, b, c\}^\star \mid |\omega|_a \neq |\omega|_b \vee |\omega|_a \neq |\omega|_c \vee |\omega|_b \neq |\omega|_c\}$.

3.12.4 Не толкова лесни задачи

Задача 3.27. Докажете, че езикът $L = \{a^n b^{kn} \mid k, n > 0\}$ не е безконтекстен.

Упътване. Да разгледаме ситуацията $\alpha = a^p b^{p^2} \in L$ и $\alpha = xyuvw$, където $y = a^i$ и $v = b^j$. Интересният случай е когато $0 < i, j < p$.

- Нека $i = j$. Да разглеждаме думата $xy^{p^2+1}uv^{p^2+1}w$, която за да бъде в езика L означава, че трябва $p + p^2 i$ дели $p^2 + p^2 i$, т.е. $1 + pi$ трябва да дели $p + pi$.

$$p + pi = k(1 + pi), \text{ за някое } 1 \leq k < p$$
$$p = k + pi(k - 1), \text{ за някое } 1 \leq k < p$$

  Достигаме до противоречие.

- Нека $i > j$, т.е. $i \geq j + 1$. Отново разглеждаме думата $xy^{p^2+1}uv^{p^2+1}w$. За да бъде тази дума в езика L, трябва $p + p^2 i$ да дели $p^2 + p^2 j$, т.е. $1 + pi$ трябва да дели $p + pj$, но

$$1 + pi \geq 1 + p(j + 1) > p + pj.$$

  Противоречие.

- Нека $i < j$ и тогава нека $j = mi + r$, където $p > i > r \geq 0$ Разглеждаме думата $xy^{mp^2+1}uv^{mp^2+1}w$. Това означава дали $p + mp^2 i$ дели $p^2 + mp^2 j$, т.е. дали $1 + mpi$ дели $p + mpj$.

$$p + pmj = k(1 + pmi), \text{ за някое } 1 \leq k.$$

  – Възможно ли е $k \geq p$? Тогава:

$$p + pmj = k(1 + pmi) \geq p(1 + pmi) \geq p(1 + pj)$$
$$p(1 + mj) \geq p(1 + pj)$$
$$m \geq p.$$

    Достигаме до противоречие. Следователно, $1 \leq k < p$.

  – Възможно ли е $k \leq m$? Тогава:

$$p + pmj = k(1 + pmi) \leq m(1 + pmi) \leq m(1 + pj)$$
$$p + pmj \leq m + pmj$$
$$p \leq m.$$

    Достигаме до противоречие, защото $m < p$.

  – Заключаваме, че $1 \leq m < k < p$. Тогава:

$$p + pmj = k(1 + pmi)$$
$$p = k + pm(ki - j).$$

    Понеже $m < k$, то $j < (m+1)i \leq ki$, т.е. $ki - j > 0$. Достигаме до противоречие.

□

Дефинираме функцията $\texttt{diff} : \Sigma^\star \times \Sigma^\star \to \Sigma^\star$ по следния начин:

$$\begin{aligned}
&\texttt{diff}(\alpha, \varepsilon) = 0 \\
&\texttt{diff}(\varepsilon, \beta) = 0 \\
&\texttt{diff}(a \cdot \alpha, b \cdot \beta) = \begin{cases} \texttt{diff}(\alpha, \beta), & \text{ако } a = b \\ 1 + \texttt{diff}(\alpha, \beta), & \text{ако } a \neq b \end{cases}
\end{aligned}$$

Задача 3.28. За всеки от следните езици, отговорете дали са безконтекстни, като се обосновете:

Да а) $\{\alpha\sharp\beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta^{\texttt{rev}}) = 1\}$;

Да б) $\{\alpha\sharp\beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta^{\texttt{rev}}) \geq 1\}$;

Не в) $\{\alpha\sharp\beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) \geq 1\}$;

Да г) $\{\alpha\beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) \geq 1\}$;

Не д) $\{\alpha\beta \mid \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) = 1\}$;

Задача 3.29. Да разгледаме езика

$$D_1 \stackrel{\text{деф}}{=} \{\alpha_1\sharp\cdots\sharp\alpha_n \mid n \geq 2 \,\&\, |\alpha_i| = |\alpha_{i+1}| \,\&\, \texttt{diff}(\alpha_i, \alpha_{i+1}^{\texttt{rev}}) = 1\}.$$

- Докажете, че D_1 не е безконтекстен.
- Докажете, че D_1 може да се представи като сечението на два безконтекстни езика.

Упътване. Да разгледаме безконтекстния език

$$L_1 = \{\alpha_1\sharp\alpha_2 \mid |\alpha_1| = |\alpha_2| \,\&\, \texttt{diff}(\alpha_1, \alpha_2^{\texttt{rev}}) = 1\}.$$

Тогава

$$\begin{aligned}
D_1 = L_1 \cdot (\sharp L_1)^\star \cdot (\{\varepsilon\} \cup \sharp\{a, b\}^\star) \cap \\
\{a, b\}^\star \cdot (\sharp L_1)^\star \cdot (\{\varepsilon\} \cup \sharp\{a, b\}^\star).
\end{aligned}$$

□

Задача 3.30. За произволен език L, дефинираме езика

$$\texttt{Diff}_n(L) = \{\alpha \in L \mid (\exists\beta \in L)[|\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) = n]\}.$$

Вярно ли е, че:

- ако L е регулярен, то $\texttt{Diff}_n(L)$ е регулярен?

- ако L е безконтекстен, то $\texttt{Diff}_n(L)$ е безконтекстен?

Задача 3.31. Нека L_1 и L_2 са езици. Дефинираме [Sip12, стр. 158]

$$L_1 \triangle L_2 \stackrel{\text{деф}}{=} \{\alpha\beta \mid \alpha \in L_1 \,\&\, \beta \in L_2 \,\&\, |\alpha| = |\beta|\}.$$

Докажете, че

а) ако L_1 и L_2 са регулярни езици, то е възможно $L_1 \triangle L_2$ да не е регулярен;

б) ако L_1 и L_2 са регулярни езици, то $L_1 \triangle L_2$ е безконтекстен;

в) ако L_1 и L_2 са безконтекстни езици, то е възможно $L_1 \triangle L_2$ да не е безконтекстен.

Задача 3.32. Докажете, че ако L е безконтекстен език, то

$$L^{\texttt{rev}} = \{\omega^{\texttt{rev}} \mid \omega \in L\}$$

също е безконтекстен.

Задача 3.33. Нека $\Sigma = \{a, b, c, d, f, e\}$. Докажете, че езикът L е безконтекстен, където за думите $\omega \in L$ са изпълнени свойствата:

- за всяко $n \in \mathbb{N}$, след всяко срещане на n последователни a-та следват n последователни b-та, и b-та не се срещат по друг повод в ω, и
- за всяко $m \in \mathbb{N}$, след всяко срещане на m последователни c-та следват m последователни d-та, и d-та не се срещат по друг повод в ω, и
- за всяко $k \in \mathbb{N}$, след всяко срещане на k последователни f-а следват k последователни e-та, и e-та не се срещат по друг повод в ω.

Задача 3.34. Да разгледаме езиците:

$$P = \{\alpha \in \{a, b, c\}^* \mid \alpha \text{ е палиндром с четна дължина}\}$$
$$L = \{\beta b^n \mid n \in \mathbb{N}, \beta \in P^n\}.$$

Да се докаже, че:

а) L не е регулярен;

б) L е безконтекстен.

Задача 3.35. Нека L_1 е произволен регулярен език над азбуката Σ, а L_2 е езика от всички думи палиндроми над Σ. Докажете, че L е безконтекстен език, където:

$$L = \{\alpha_1 \cdots \alpha_{3n}\beta_1 \cdots \beta_m\gamma_1 \cdots \gamma_n \mid \alpha_i, \gamma_j \in L_1, \beta_k \in L_2, m, n \in \mathbb{N}\}.$$

Задача 3.36. Нека $L = \{\omega \in \{a, b\}^\star \mid |\omega|_a = 2\}$. Да се докаже, че езикът $L' = \{\alpha^n \mid \alpha \in L, n \geq 0\}$ не е безконтекстен.

Задача 3.37. Нека $\Sigma = \{a, b, c\}$ и $L \subseteq \Sigma^\star$ е безконтестен език. Ако имаме дума $\alpha \in \Sigma^\star$, тогава *L-вариант* на α ще наричаме думата, която се получава като в α всяко едно срещане на символа a заменим с (евентуално различна) дума от L. Тогава, ако $M \subseteq \Sigma^*$ е произволен безконтестен език, да се докаже че езикът

$$M' = \{\beta \in \Sigma^\star | \ \beta \text{ е } L\text{-вариант на } \alpha \in M\}$$

също е безконтекстен.

Задача 3.38. Докажете, че всеки безконтекстен език над азбуката $\Sigma = \{a\}$ е регулярен.

Задача 3.39. Да фиксираме азбуката Σ. Нека L е безконтекстен език, а R е регулярен език. Докажете, че езикът

$$L/_R = \{\alpha \in \Sigma^\star \mid (\exists \beta \in R)[\alpha\beta \in L]\}$$

е безконтекстен.

Упътване. Най-лесно се вижда с декартова конструкция на стеков автомат за L и автомат за R. □

Задача 3.40. Нека е дадена граматиката $G = \langle\{a, b\}, \{S, A, B, C\}, S, R\rangle$. Използвайте CYK-алгоритъма, за да проверите дали думата α принадлежи на $\mathcal{L}(G)$, където правилата на граматиката и думата α са зададени като:

а) $S \to BA \mid CA \mid a,\ C \to BS \mid SA,\ A \to a,\ B \to b$,
$\alpha = bbaaa$;

б) $S \to AB \mid BC,\ A \to BA \mid a,\ B \to CC \mid b,\ C \to AB \mid a$,
$\alpha = baaba$;

в) $S \to AB,\ A \to AC \mid a \mid b,\ B \to CB \mid a,\ C \to a$,
$\alpha = babaa$.

Задача 3.41. Нека L е безконтекстен език над азбуката Σ. Докажете, че следните езици са безконтекстни:

а) $\texttt{Pref}(L) = \{\alpha \in \Sigma^\star \mid (\exists \beta \in \Sigma^\star)[\alpha \cdot \beta \in L]\}$;

б) $\texttt{Suff}(L) = \{\beta \in \Sigma^\star \mid (\exists \alpha \in \Sigma^\star)[\alpha \cdot \beta \in L]\}$;

в) $\texttt{Infix}(L) = \{\beta \in \Sigma^\star \mid (\exists \alpha, \gamma \in \Sigma^\star)[\alpha \cdot \beta \cdot \gamma \in L]\}$;

Задача 3.42. Докажете, че езикът

$$L = \{\ \omega_1\sharp\omega_2\sharp\cdots\sharp\omega_{2n} \mid n \geq 1\ \&\ \sum_{i=1}^{n} |\omega_{2i-1}| = \sum_{i=1}^{n} |\omega_{2i}|\ \}$$

е безконтекстен.

Упътване. Най-лесно става със стеков автомат, като са необходими две допълнителни букви за азбуката на стека.

Една възможна безконтекстна граматика е следната:

$$
\begin{aligned}
&E \to XEX \mid \sharp O \mid O\sharp \mid \sharp E\sharp \\
&O \to OO \mid EE \mid \varepsilon \\
&X \to a \mid b,
\end{aligned}
$$

където началната променлива е E. □

Глава 4

Машини на Тюринг

> Turing's 'Machines'. These machines are humans who calculate.
> [WWA80, § 1096].

Детерминистична машина на Тюринг ще наричаме осморка от вида

$$\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, \sqcup, q_{\texttt{start}}, q_{\texttt{accept}}, q_{\texttt{reject}} \rangle,$$

Тук до голяма степен следваме [Sip12, Глава 3]. Понятието за машина на Тюринг има много еквивалентни дефиниции.

където:

- Q - крайно множество от състояния;
- Σ - крайна азбука за входа;
- Γ - крайна азбука за лентата, $\Sigma \subseteq \Gamma$;
- $\sqcup$ - символ за празна клетка на лентата, $\sqcup \in \Gamma \setminus \Sigma$;
- $q_{\texttt{start}} \in Q$ - начално състояние;
- $q_{\texttt{accept}} \in Q$ - приемащо състояние;
- $q_{\texttt{reject}} \in Q$ - отхвърлящо състояние, където $q_{\texttt{accept}} \neq q_{\texttt{reject}}$;
- $\delta : Q' \times \Gamma \to Q \times \Gamma \times \{\triangleleft, \triangleright, \square\}$ - тотална функция на преходите, където $Q' = Q \setminus \{q_{\texttt{accept}}, q_{\texttt{reject}}\}$.

Тези две състояния ще наричаме заключителни

Това означава, че веднъж достигнем ли заключително състояние, не можем да правим повече преходи. Тук следваме [Sip12, стр. 169] и [HMU01, стр. 327].

Всяка машина на Тюринг разполага с неограничено количество памет, която е представена като безкрайна (и в двете посоки) лента, разделена на клетки. Всяка клетка съдържа елемент на Γ. Сега ще опишем как $\mathcal{M}$ работи върху вход думата $\alpha \in \Sigma^\star$. Първоначално безкрайната лента съдържа само думата α. Останалите клетки на лентата съдържат символа $\sqcup$.

Освен това, $\mathcal{M}$ се намира в началното състояние $q_{\texttt{start}}$ и главата за четене е върху най-левия символ на α. Работата на $\mathcal{M}$ е описана от функцията на преходите δ.

На англ. instanteneous description. Понякога за удобство ще означаваме моментната конфигурация като $(q, \alpha\underline{x}\beta)$ вместо по-неудобното $(\alpha, q, x\beta)$.

- Формално, **моментната конфигурация** (или описание) на едно изчисление на машина на Тюринг е тройка от вида

$$(\alpha, q, \beta) \in \Gamma^\star \times Q \times \Gamma^+,$$

като интерпретацията на тази тройка е, че машината се намира в състояние q и лентата има вида

$$\cdots \sqcup \sqcup \sqcup \, \alpha\underline{x}\beta' \, \sqcup \sqcup \sqcup \cdots,$$

където $\beta = x\beta'$ и четящата глава на машината е поставена върху x.

- Макар и да имаме безкрайна лента, моментната конфигурация, която може да се представи като *крайна* дума, описва цялото моментно състояние на машината на Тюринг.

- **Началната конфигурация** за входната дума $\alpha \in \Sigma^\star$ представлява тройката

$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup).$$

- **Приемаща конфигурация** представлява тройка от вида

$$(\beta, q_{\texttt{accept}}, \gamma).$$

- **Отхвърляща конфигурация** представлява тройка от вида

$$(\beta, q_{\texttt{reject}}, \gamma).$$

- Една конфигурация ще наричаме **заключителна**, ако тя е или приемаща или отхвърляща.

Както за автомати, удобно е да дефинираме бинарна релация $\vdash$ над $\Gamma^\star \times Q \times \Gamma^+$, която ще казва как моментната конфигурация на машината $\mathcal{M}$ се променя при изпълнение на една стъпка.

Ако няма опасност да се заблудим за коя точно машина на Тюринг $\mathcal{M}$ говорим, то е възможно да пишем просто $\vdash$.

$$\frac{\delta(q,x) = (p,y,\rhd)}{(\alpha,q,xz\beta) \vdash_{\mathcal{M}} (\alpha y,p,z\beta)} \text{ (right-1)} \qquad \frac{\delta(q,x) = (p,y,\lhd)}{(\alpha z,q,x\beta) \vdash_{\mathcal{M}} (\alpha,p,zy\beta)} \text{ (left-1)}$$

$$\frac{\delta(q,x) = (p,y,\rhd)}{(\alpha,q,x) \vdash_{\mathcal{M}} (\alpha y,p,\sqcup)} \text{ (right-2)} \qquad \frac{\delta(q,x) = (p,y,\lhd)}{(\varepsilon,q,x\beta) \vdash_{\mathcal{M}} (\varepsilon,p,\sqcup y\beta)} \text{ (left-2)}$$

$$\frac{\delta(q,z) = (p,y,\square)}{(\alpha,q,z\beta) \vdash_{\mathcal{M}} (\alpha,p,y\beta)} \text{ (stay)}$$

Фигура 4.1: Едностъпков преход в детерминистична машина на Тюринг

Сега за всяко естествено число ℓ, ще дефинираме релацията $\vdash^\ell$, която ще казва, че от конфигурацията κ можем да достигнем до конфигурацията κ' за ℓ на брой стъпки.

$$\frac{}{\kappa \vdash^0 \kappa} \text{ (рефлексивност)} \qquad \frac{\kappa \vdash \kappa'' \qquad \kappa'' \vdash^\ell \kappa'}{\kappa \vdash^{\ell+1} \kappa'} \text{ (транзитивност)}$$

- С $\vdash^\star$ ще означаваме рефлексивното и транзитивно затваряне на релацията $\vdash$ или с други думи,
$$\kappa \vdash^\star \kappa' \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[\kappa \vdash^\ell \kappa'].$$
- Макар и една конфигурация κ да преставлява тройка, то често ще бъде удобно да гледаме на κ като на дума от $\Gamma^\star Q \Gamma^+$.
- Важно свойство е, че ако $\kappa \vdash^\star \kappa'$, то $|\kappa| \leq |\kappa'|$.
- машината на Тюринг $\mathcal{M}$ **приема** думата α, ако за някои $\lambda, \rho \in \Gamma^\star$,
$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash^\star_{\mathcal{M}} (\lambda, q_{\texttt{accept}}, \rho).$$
Важно е да имаме $\sqcup$ след думата α, защото е възможно α да е празната дума.
- Машината на Тюринг $\mathcal{M}$ **отхвърля** думата α, ако за някои $\lambda, \rho \in \Gamma^\star$,
$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash^\star_{\mathcal{M}} (\lambda, q_{\texttt{reject}}, \rho).$$
- Машината на Тюринг $\mathcal{M}$ **не приема** думата α, ако $\mathcal{M}$ отхвърля α или $\mathcal{M}$ никога не завършва при начална конфигурация $(\varepsilon, q_{\texttt{start}}, \alpha)$.
- Една машина на Тюринг се нарича **разрешител**, ако при всеки вход достига до заключително състояние, т.е. достига до $q_{\texttt{accept}}$ или $q_{\texttt{reject}}$.

На англ. такава машина на Тюринг се нарича **decider** [Sip12, стр. 170]. Може такива машини на Тюринг да се наричат и тотални [Koz97, стр. 213]. Да се внимава, че в Манев понятията са различни.

- Езикът, който се **разпознава** от машината $\mathcal{M}$ е:
$$\mathcal{L}(\mathcal{M}) \overset{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid (\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash^\star_{\mathcal{M}} (\lambda, q_{\texttt{accept}}, \rho), \text{ за някои } \lambda, \rho \in \Gamma^\star\}.$$
- Езикът L се нарича **полуразрешим**, ако съществува машина на Тюринг $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$. В този случай се казва, че $\mathcal{M}$ разпознава езика L. Ако една дума $\alpha \in L$, то след крайно много стъпки ще достигнем до състоянието $q_{\texttt{accept}}$. Ако $\alpha \notin L$, то не е ясно дали какво се случва с изчислението на $\mathcal{M}$ върху α. Възможно е да достигнем до състоянието $q_{\texttt{reject}}$, но може да попаднем в безкрайно изчисление.

На англ. **semidecidable language**. В литературата се използва и названието **рекурсивно номеруем език**.

- Един език L се нарича **разрешим**, ако за него съществува *разрешител* $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$. В този случай се казва, че $\mathcal{M}$ разрешава езика L.

На англ. **decidable language**. В литературата се използва и названието **рекурсивен език**.

Твърдение 4.1. Ако L е разрешим език над азбуката Σ, то $\Sigma^\star \setminus L$ също е разрешим език.

От дефинициите е ясно, че всеки разрешим език е полуразрешим. По-късно, ще видим, че съществуват полуразрешими езици, чиито допълнения не са полуразрешими, т.е. не всеки полуразрешим език е разрешим. Една от основните ни задачи ще бъде да класифицираме различни езици като (не)разрешими и (не)полуразрешими. За да придобием по-добра интуиция за тези нови понятия, ще разгледаме подробно няколко примера. Ще видим също как можем да изобразяваме функцията на преходите на $\mathcal{M}$ графично.

Примери

Пример 4.1. Да разгледаме езика $L = \{a^n b^n c^n \mid n \in \mathbb{N}\}$. Нека да видим защо този език е разрешим. Идеята на алгоритъма, който ще разгледаме е да маркира на всяка итерация по един символ a, b или c. Той завършва успешно, ако всички символи на думата са маркирани. Да въведем нов символ x, с който ще маркираме обработените от входната дума символи a, b, c. Нека първоначално думата е копирана върху лентата и четящата глава е върху първия символ на думата.

Знаем, че L не е безконтекстен, но е контекстен.

Алгоритъмът има времева сложност $\mathcal{O}(n^2)$. Ако разгледаме машина на Тюринг с три ленти, то ще получим времева сложност $\mathcal{O}(n)$.

(1) Чете x-ове надясно по лентата докато срещне първото a и го замества с x. Отива на стъпка (2). Ако символите свършат (т.е. достигне се $\sqcup$) преди да се достигне a, то алгоритъмът завършва успешно.

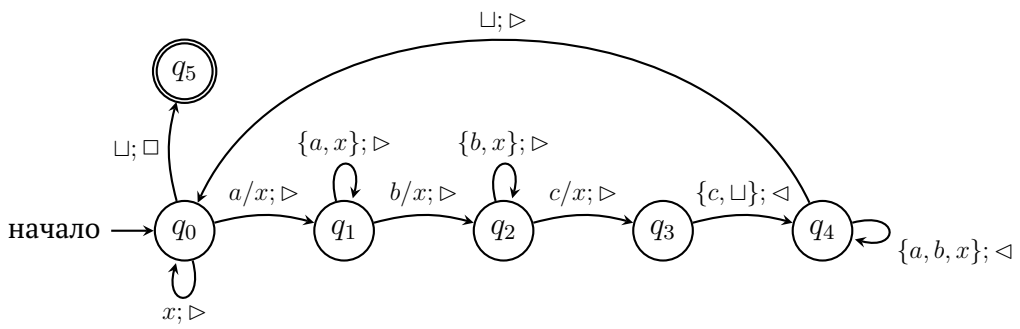
(2) Чете x-ове надясно по лентата докато срещне първото b и го замества с x. Отива на стъпка (3).

(3) Чете x-ове надясно по лентата докато срещне първото c и го замества с x.

(4) Връща четящата глава в началото на лентата, т.е. чете наляво докато не срещне символа $\sqcup$. Връща се в стъпка (1).

Нека сега да видим, че този алгоритъм може да се опише съвсем формално с машина на Тюринг. Ще построим машина на Тюринг $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$, където

- $\Sigma = \{a, b, c\}$;
- $\Gamma = \{a, b, c, x, \sqcup\}$, за някой нов символ x;
- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$, като $q_{\texttt{start}} = q_0$ и $q_{\texttt{accept}} = q_5$;
- Частичната функция на преходите $\delta : (Q \setminus \{q_5\}) \times \Gamma \to Q \times \Gamma \times \{\triangleleft, \triangleright, \square\}$ е описана на схемата отдолу. Остава да добавим състоянието $q_{\texttt{reject}}$.

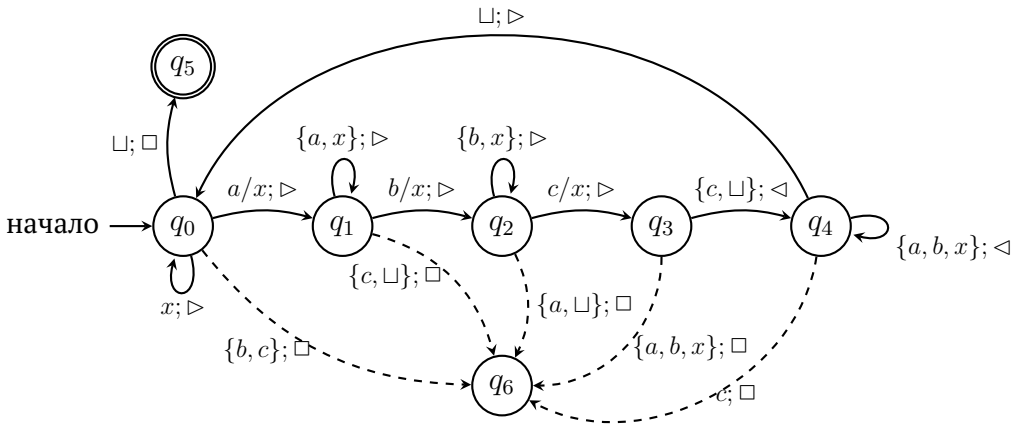


Фигура 4.3: Част от детерминистична машина на Тюринг $\mathcal{M}$, за която разрешава езика $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. Остава да се добави $q_{\texttt{reject}}$ и да се попълни δ функцията до тотална.

Горната схема определя точно функцията на преходите δ. Например,

$$\delta(q_0, a) = (q_1, x, \triangleright)$$
$$\delta(q_4, \sqcup) = (q_0, \sqcup, \triangleright)$$
$$\delta(q_1, a) = (q_1, a, \triangleright)$$
$$\delta(q_1, x) = (q_1, x, \triangleright).$$

Съобразете, че тази машина на Тюринг може да се направи тотална като се добави ново състояние $q_6 = q_{\texttt{reject}}$ и за всяка двойка (q, z), за която функцията на преходите не е дефинирана, да сочи към $q_{\texttt{reject}}$. Така получаваме пълното описание на детерминистична машина на Тюринг $\mathcal{M}$, за която $\mathcal{L}(\mathcal{M}) = L$. Лесно се съобразява, че тази машина на Тюринг е *тотална*, т.е. за всеки вход $\mathcal{M}$ завършва в $q_{\texttt{accept}}$ или $q_{\texttt{reject}}$. Заключаваме, че L е не само полуразрешим, но *разрешим* език.



Фигура 4.4: Детерминистична машина на Тюринг $\mathcal{M}$, която *разрешава* езика $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. Отхвърлящото състояние е q_6, т.е. $q_{\texttt{reject}} \stackrel{\text{деф}}{=} q_6$.

Да напомним, че този език не е безконтекстен. В [HU79, стр. 155] е дадено по-различно решение. Тук следваме [Sip12, стр. 173]. Там има малка грешка. Това запаметяване става в състоянията.

Пример 4.2. Да разгледаме езика $L = \{\omega\sharp\omega \mid \omega \in \{a, b\}^\star\}$. Първо неформално ще опишем алгоритъм, който да разпознава думите на езика L. Нека една дума е копирана върху лентата и четящата глава е поставена върху първия символ от думата.

(1) Чете x-ове надясно по лентата докато не срещне a или b и го замества с x. Запомня дали сме срещнали a или b. Ако вместо a или b срещне $\sharp$, то отива на стъпка (6).

(2) Чете a-та и b-та надясно по лентата докато не стигне $\sharp$.

(3) Чете символа $\sharp$ надясно по лентата и всички следващи x-ове докато не срещне символа a или b. Той трябва да е същия символ, който сме запаметили на стъпка (1). Заместваме този символ с x.

(4) Чете x-ове наляво по лентата докато не стигне $\sharp$.

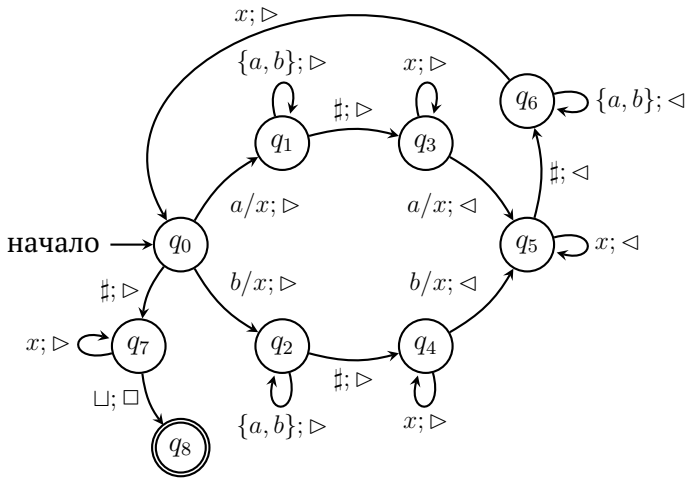
(5) Чете a-та и b-та по лентата докато не стигне x. Поставя четящата глава върху символа точно след първия x. Отива на стъпка (1).

(6) Прочита $\sharp$ надясно по лентата и чете надясно x-ове докато не срещне $\sqcup$. Алгоритъмът завършва успешно.

Обърнете внимание, че този алгоритъм има времева сложност $\mathcal{O}(n^2)$. Ще видим в Пример 4.3, че ако разгледаме двулентова машина на Тюринг, то имаме алгоритъм със сложност $\mathcal{O}(n)$.

Ще построим машина на Тюринг $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$.

- $\Sigma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp, x, \sqcup\}$;
- $Q \stackrel{\text{деф}}{=} \{q_0, q_1, \ldots, q_8\}$, като $q_{\texttt{start}} \stackrel{\text{деф}}{=} q_0$ и $q_{\texttt{accept}} \stackrel{\text{деф}}{=} q_8$;



Фигура 4.5: Част от машина на Тюринг $\mathcal{M}$, която разрешава езика $\{\omega\sharp\omega \mid \omega \in \{a,b\}^\star\}$.

Да проследим изчислението на думата $ab\sharp ab$.

$$\begin{aligned}(q_0, \underline{a}b\sharp ab\sqcup) &\vdash_{\mathcal{M}} (q_1, x\underline{b}\sharp ab\sqcup) \vdash_{\mathcal{M}} (q_1, xb\underline{\sharp}ab\sqcup) \vdash_{\mathcal{M}} (q_3, xb\sharp\underline{a}b\sqcup)\\ &\vdash_{\mathcal{M}} (q_5, xb\underline{\sharp}xb\sqcup) \vdash_{\mathcal{M}} (q_6, x\underline{b}\sharp xb\sqcup) \vdash_{\mathcal{M}} (q_6, \underline{x}b\sharp xb\sqcup)\\ &\vdash_{\mathcal{M}} (q_0, x\underline{b}\sharp xb\sqcup) \vdash_{\mathcal{M}} (q_2, xx\underline{\sharp}xb\sqcup) \vdash_{\mathcal{M}} (q_4, xx\sharp\underline{x}b\sqcup)\\ &\vdash_{\mathcal{M}} (q_4, xx\sharp x\underline{b}\sqcup) \vdash_{\mathcal{M}} (q_5, xx\sharp\underline{x}x\sqcup) \vdash_{\mathcal{M}} (q_5, xx\underline{\sharp}xx\sqcup)\\ &\vdash_{\mathcal{M}} (q_6, x\underline{x}\sharp xx\sqcup) \vdash_{\mathcal{M}} (q_0, xx\underline{\sharp}xx\sqcup) \vdash_{\mathcal{M}} (q_7, xx\sharp\underline{x}x\sqcup)\\ &\vdash_{\mathcal{M}} (q_7, xx\sharp x\underline{x}\sqcup) \vdash_{\mathcal{M}} (q_7, xx\sharp xx\underline{\sqcup}) \vdash_{\mathcal{M}} (q_8, xx\sharp xx\underline{\sqcup}).\end{aligned}$$

Може лесно да се съобрази, че тази машина на Тюринг може да се допълни до *тотална*.

Задача 4.1. Докажете, че езикът $L = \{\, \omega \cdot \omega \mid \omega \in \{a,b\}^\star \,\}$ е разрешим.

4.1 Многолентови детерминирани машини на Тюринг

Детерминирана машина на Тюринг с k ленти има същата дефиниция като еднолентова машина на Тюринг с единствената разлика, че

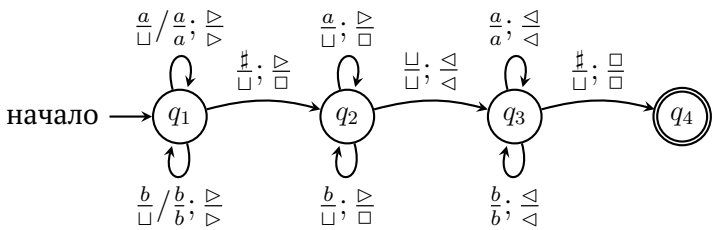
$$\delta : Q' \times \Gamma^k \to Q \times \Gamma^k \times \{\triangleleft, \triangleright, \square\}^k,$$

където $Q' = Q \setminus \{q_{\texttt{accept}}, q_{\texttt{reject}}\}$, т.е. имаме k на брой четящи глави, по една за всяка лента, които се движат независимо една от друга. Приемаме, че входната дума α е записана върху първата лента.

Пример 4.3. Да видим как двулентова машина на Тюринг разрешава езика

$$L = \{\, \omega\sharp\omega \mid \omega \in \{a, b\}^\star \,\}.$$

- $Q = \{q_1, q_2, q_3, q_4, q_5\}$;
- $q_{\texttt{start}} = q_1$, $q_{\texttt{accept}} = q_4$ и $q_{\texttt{reject}} = q_5$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp, \sqcup\}$;
- $\delta : Q' \times \Gamma^2 \to Q \times \Gamma^2 \times \{\triangleleft, \triangleright, \square\}^2$, където $Q' = \{q_1, q_2, q_3\}$.



Фигура 4.6: Непълно описание на двулентова детерминирана машина на Тюринг $\mathcal{M}$, за която $\mathcal{L}(\mathcal{M}) = \{\omega\sharp\omega \mid \omega \in \{a, b\}^\star\}$. Всички неописани преходи трябва да сочат към отхвърлящото състояние q_5.

В началото втората лента е празна. Имаме две глави, които се движат независимо една от друга. При вход думата $\omega\sharp\omega$, $\mathcal{M}$ първо копира ω върху втората лента. След това сравнява това, което е записано на втората лента с думата, която следва след символа $\sharp$. Лесно се съобразява, че сега сложността на изчислението е $\mathcal{O}(n)$, докато при еднолентова машина на Тюринг то беше $\mathcal{O}(n^2)$. Да разгледаме

един пример:

$$
\begin{aligned}
(q_1, \tfrac{\hat{a}}{\hat{\sqcup}}\tfrac{b}{\sqcup}\tfrac{\sharp}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) &\vdash (q_1, \tfrac{a}{a}\tfrac{\hat{b}}{\hat{\sqcup}}\tfrac{\sharp}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_1, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\hat{\sharp}}{\hat{\sqcup}}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \\
&\vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{\hat{a}}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{\hat{a}}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \\
&\vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{a}{\sqcup}\tfrac{\hat{b}}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\hat{\sqcup}}{\sqcup}) \\
&\vdash (q_3, \tfrac{a}{a}\tfrac{b}{\hat{b}}\tfrac{\sharp}{\sqcup}\tfrac{a}{\sqcup}\tfrac{\hat{b}}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_3, \tfrac{a}{\hat{a}}\tfrac{b}{b}\tfrac{\sharp}{\sqcup}\tfrac{\hat{a}}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \\
&\vdash (q_3, \tfrac{\sqcup}{\hat{\sqcup}}\tfrac{a}{a}\tfrac{b}{b}\tfrac{\hat{\sharp}}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_4, \tfrac{\sqcup}{\hat{\sqcup}}\tfrac{a}{a}\tfrac{b}{b}\tfrac{\hat{\sharp}}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}).
\end{aligned}
$$

Твърдение 4.2. За всяка k-лентова машина на Тюринг $\mathcal{M}$ съществува еднолентова машина на Тюринг $\mathcal{M}'$, такава че $\mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{M}')$.

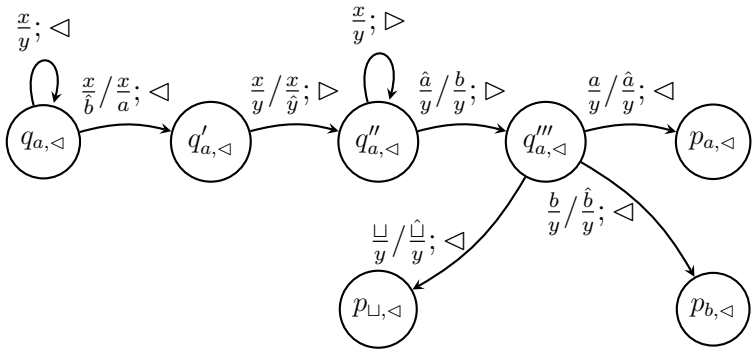
Упътване. Нека $\mathcal{M}$ е k-лентова машина на Тюринг. Ще построим еднолентова машина на Тюринг $\mathcal{M}'$, за която $\mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{M}')$. Да означим $\hat{\Gamma} = \{\hat{x} \mid x \in \Gamma\}$. Тогава азбуката на лентата на $\mathcal{M}'$ ще бъде $\Gamma' = \Sigma \cup (\hat{\Gamma} \cup \Gamma)^k$. Сега вместо да имаме k ленти със символи от Γ, ще имаме една лента със символи k-орки от $\hat{\Gamma} \cup \Gamma$. За да симулираме $\mathcal{M}$, използваме символите $\hat{x}$ за да маркират позицията на главите на $\mathcal{M}$, като във всяка компонента на лентата има точно по един символ от вида $\hat{x}$. При вход думата $\alpha = a_1 a_2 \cdots a_n$, която е поставена върху единствената лента на $\mathcal{M}'$, в случая на $k = 2$, първата работа на $\mathcal{M}'$ е да замени α с думата

В [Sip12, стр. 177] конструкцията е малко по-различна. Там съдържанието на всяка лента се поставя последователно върху една лента, като се разделят със специален символ. Тук следваме [HU79, стр. 162].

$$
\frac{\hat{a}_1}{\hat{\sqcup}}\frac{a_2}{\sqcup}\cdots\frac{a_n}{\sqcup}
$$

За да определим следващия ход на машината $\mathcal{M}'$, трябва да сканираме лентата докато не открием разположението на всичките k на брой маркирани клетки. Тогава симулираме ход на $\mathcal{M}$ и отново трябва да променим маркираните клетки. □

Да видим по-подробно как можем да симулираме изчислението на двулентова машина на Тюринг $\mathcal{M}$ с еднолентовата машина на Тюринг $\mathcal{M}'$.



Фигура 4.7: Симулация на прехода $\delta_{\mathcal{M}}(q, \frac{a}{b}) = (p, \frac{b}{a}, \frac{\triangleright}{\triangleleft})$ при положение, че главата на втората лента е наляво спрямо главата на първата лента

- В еднолентовата машина на Тюринг, за удобство пишем $\frac{x}{y}$ вместо наредената двойка (x, y).

✍ Помислете как да обобщите тази идея за машина на Тюринг с n ленти. Важното е, че във всяко състояние кодираме крайна информация.

- За всяко състояние q на $\mathcal{M}$ и всяко a от Γ, в машината $\mathcal{M}'$ ще имаме състояния от вида $q_{a,\triangleleft}, q_{a,\triangleright}, q_{a,\square}$, които носят информацията, че в състояние q на симулираната двулентова машина на Тюринг $\mathcal{M}$, главата на първата лента е върху символа a и главата на втората лента е разположена наляво/надясно/на същата позиция спрямо главата на първата лента.

- Във Фигура 4.7, $\frac{x}{y}$ е съкратен запис за $\{\frac{x}{y} \mid x, y \in \Gamma\}$;

- Възможно е на всяка симулирана стъпка, главите на двете ленти да се раздалечават. Това означава, че след симулацията на s стъпки, в най-лошия случай, двете глави са на разстояние $2s$. Тогава за да симулираме $(s+1)$-вата стъпка на $\mathcal{M}$, първо трябва да отидем $2s$ стъпки наляво, после $2s$ стъпки надясно. Следователно $(s+1)$-вата стъпка на $\mathcal{M}$ се симулира за приблизително $4s$ стъпки.

- Ако изчислението на $\mathcal{M}$ върху вход α отнема s стъпки, то симулираното изчисление върху α ще отнеме в най-лошия случай приблизително

$$\sum_{i=0}^{s} 4i = 2s^2 + 2s$$

стъпки. Заключаваме, че за s стъпки от изчисленито на $\mathcal{M}$, симулацията върху $\mathcal{M}'$ отнема време $\mathcal{O}(s^2)$, т.е. имаме квадратично забавяне.

4.2 Изчислими функции

Една *тотална* функция $f : \Sigma^\star \to \Sigma^\star$ се нарича изчислима с машина на Тюринг $\mathcal{M}$, ако за всяка дума $\alpha \in \Sigma^\star$,

$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash^\star_{\mathcal{M}} (\sqcup^m, q_{\texttt{accept}}, f(\alpha)\sqcup^k), \text{ за някои } m, k \in \mathbb{N}.$$

Възможно е да се дефинира и за двулентова машина на Тюринг, като $f(\alpha)$ ще бъде върху втората лента.

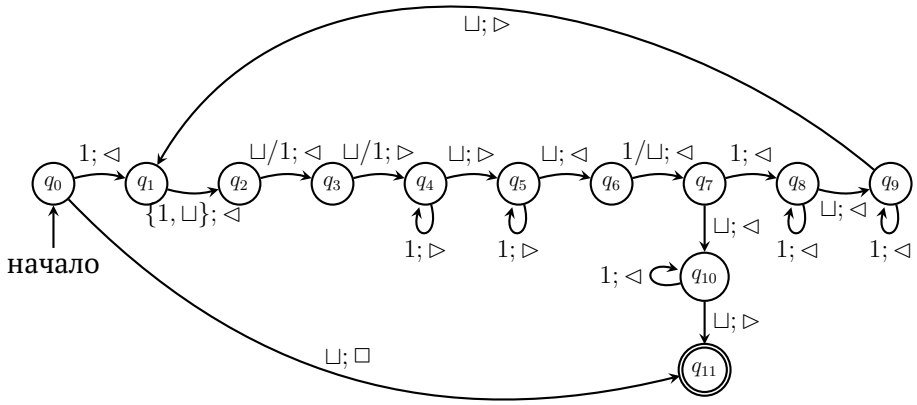
Това означава, че машината на Тюринг $\mathcal{M}$ винаги завършва. Лесно се съобразява, че езикът $\texttt{Graph}(f) = \{\, \alpha\sharp f(\alpha) \mid \alpha \in \Sigma^\star \,\}$ е разрешим.

Задача 4.2. Докажете, че съществуват функции от вида $f : \Sigma^\star \to \Sigma^\star$, които не са изчислими с машина на Тюринг.

Упътване. Всяка машина на Тюринг може да се кодира с естествено число. Това означава, че съществуват изброимо безкрайно много машини на Тюринг. От друга страна, съществуват неизброимо много функции от вида $f : \Sigma^\star \to \Sigma^\star$. □

Пример 4.4. Да разгледаме функцията $f : \{1\}^\star \to \{1\}^\star$, където $f(1^n) \stackrel{\text{деф}}{=} 1^{2n}$. Да видим защо f е изчислима с машина на Тюринг.

При двулентова машина на Тюринг тази задача е много по-лесна и сложността ще бъде $\mathcal{O}(n)$ вместо $\mathcal{O}(n^2)$.

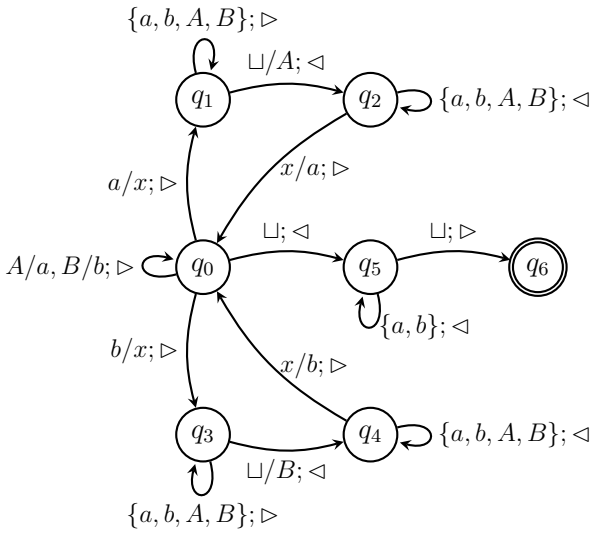


Фигура 4.8: Машина на Тюринг за функцията $f(1^n) = 1^{2n}$.

Пример 4.5. Да видим защо тоталната функция $f : \{a, b\}^\star \to \{a, b\}^\star$, дефинирана като $f(\alpha) = \alpha \cdot \alpha$ е изчислима с машина на Тюринг.

Това пак става много по-лесно с двулентова машина на Тюринг и сложността пак ще бъде $\mathcal{O}(n)$ вместо $\mathcal{O}(n^2)$.

- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{a, b, x, A, B\}$;
- $q_{\texttt{start}} \stackrel{\text{деф}}{=} q_0$ и $q_{\texttt{accept}} \stackrel{\text{деф}}{=} q_6$



Фигура 4.9: Машина на Тюринг за $f(\alpha) = \alpha \cdot \alpha$.

Да проследим работата на $\mathcal{M}$ върху думата ab. Първо копираме добавяме AB и така лентата съдържа $abAB$. След това заменяме A с a и B с b. Така най-накравя полуваме върху лентата думата $abab$.

$$\begin{aligned}(q_0, \underline{a}b\sqcup) &\vdash_{\mathcal{M}} (q_1, x\underline{b}\sqcup) \vdash_{\mathcal{M}} (q_1, xb\underline{\sqcup}) \vdash_{\mathcal{M}} (q_2, x\underline{b}A) \vdash_{\mathcal{M}} (q_2, \underline{x}bA)\\ &\vdash_{\mathcal{M}} (q_0, a\underline{b}A) \vdash_{\mathcal{M}} (q_3, ax\underline{A}) \vdash_{\mathcal{M}} (q_3, axA\underline{\sqcup}) \vdash_{\mathcal{M}} (q_4, ax\underline{A}B)\\ &\vdash_{\mathcal{M}} (q_4, a\underline{x}AB) \vdash_{\mathcal{M}} (q_0, ab\underline{A}B) \vdash_{\mathcal{M}} (q_0, aba\underline{B}) \vdash_{\mathcal{M}} (q_0, abab\underline{\sqcup})\\ &\vdash_{\mathcal{M}} (q_5, aba\underline{b}) \vdash_{\mathcal{M}} (q_5, ab\underline{a}b) \vdash_{\mathcal{M}} (q_5, a\underline{b}ab) \vdash_{\mathcal{M}} (q_5, \underline{a}bab)\\ &\vdash_{\mathcal{M}} (q_5, \underline{\sqcup}abab) \vdash_{\mathcal{M}} (q_6, \underline{a}bab).\end{aligned}$$

Не можем директно да започнем да копираме α, защото така няма да знаем къде е края на първото копие на α. Това можем да направим като първо запишем на лентата $\alpha\sharp\alpha$ и след това второто копие на α го изместим с една позиция наляво.

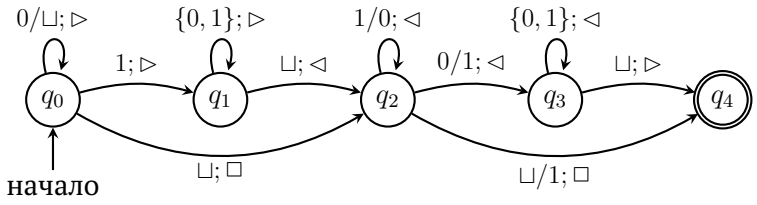
Изискваме $f(\alpha)$ да започва с 1 за да може f да бъде функция, т.е. $f(\alpha)$ е най-късият двоичен запис на числото $\overline{\alpha}_{(2)} + 1$.

Пример 4.6. Да разгледаме тоталната функция $f : \{0,1\}^\star \to 1 \cdot \{0,1\}^\star$, където

$$\overline{f(\alpha)}_{(2)} \stackrel{\text{деф}}{=} \overline{\alpha}_{(2)} + 1.$$

Нека да видим, че тази функция е изчислима с машина на Тюринг.

- $\Sigma \stackrel{\text{деф}}{=} \{0,1\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{0,1,\sqcup\}$;
- $q_{\texttt{start}} = q_0$ и $q_{\texttt{accept}} \stackrel{\text{деф}}{=} q_4$.



Фигура 4.10: Машина на Тюринг изчисляваща f, за която $\overline{f(\alpha)}_{(2)} = \overline{\alpha}_{(2)} + 1$.

Да проследим изчислението на $\mathcal{M}$ върху вход 01011.

$$\begin{aligned}(q_0, 0\underline{1}011\sqcup) &\vdash_{\mathcal{M}} (q_0, \sqcup\underline{1}011\sqcup) \vdash_{\mathcal{M}} (q_1, \sqcup1\underline{0}11\sqcup) \vdash_{\mathcal{M}} (q_1, \sqcup10\underline{1}1\sqcup)\\ &\vdash_{\mathcal{M}} (q_1, \sqcup1011\underline{\sqcup}) \vdash_{\mathcal{M}} (q_1, \sqcup1011\underline{\sqcup}) \vdash_{\mathcal{M}} (q_2, \sqcup101\underline{1}\sqcup)\\ &\vdash_{\mathcal{M}} (q_2, \sqcup10\underline{1}0\sqcup) \vdash_{\mathcal{M}} (q_2, \sqcup1\underline{0}00\sqcup) \vdash_{\mathcal{M}} (q_3, \sqcup\underline{1}100\sqcup)\\ &\vdash_{\mathcal{M}} (q_3, \underline{\sqcup}1100\sqcup) \vdash_{\mathcal{M}} (q_4, \sqcup\underline{1}100\sqcup).\end{aligned}$$

Задача 4.3. Да разгледаме азбуката $\Sigma = \{0, 1, \ldots, k-1\}$, където $k > 2$. Да разгледаме тоталната функция

$$f : \Sigma^\star \to (\Sigma \setminus \{0\}) \cdot \Sigma^\star,$$

дефинирана като

$$\overline{f(\alpha)}_{(k)} = \overline{\alpha}_{(k)} + 1.$$

Дефинирайте машина на Тюринг $\mathcal{M}$, която изчислява функцията f.

4.3 Недетерминирани машини на Тюринг

Една машина на Тюринг $\mathcal{N}$ се нарича недетерминирана, ако функцията на преходите има вида

$$\Delta : Q' \times \Gamma \to \mathscr{P}(Q \times \Gamma \times \{\triangleleft, \triangleright, \square\}),$$

където да напомним, че $Q' = Q \setminus \{q_{\texttt{accept}}, q_{\texttt{reject}}\}$.

Отново можем да дефинираме бинарна релация $\vdash$ над $\Gamma^\star \times Q \times \Gamma^+$, която ще казва как моментното описание на машината $\mathcal{N}$ се променя при изпълнение на една стъпка.

$$\frac{\Delta(q,x) \ni (p,y,\triangleright)}{(\alpha,q,xz\beta) \vdash_{\mathcal{N}} (\alpha y,p,z\beta)}\ \text{(right-1)} \qquad \frac{\Delta(q,x) \ni (p,y,\triangleleft)}{(\alpha z,q,x\beta) \vdash_{\mathcal{N}} (\alpha,p,zy\beta)}\ \text{(left-1)}$$

$$\frac{\Delta(q,x) \ni (p,y,\triangleright)}{(\alpha,q,x) \vdash_{\mathcal{N}} (\alpha y,p,\sqcup)}\ \text{(right-2)} \qquad \frac{\Delta(q,x) \ni (p,y,\triangleleft)}{(\varepsilon,q,x\beta) \vdash_{\mathcal{N}} (\varepsilon,p,\sqcup y\beta)}\ \text{(left-2)}$$

$$\frac{\Delta(q,z) \ni (p,y,\square)}{(\alpha,q,z\beta) \vdash_{\mathcal{N}} (\alpha,p,y\beta)}\ \text{(stay)}$$

Фигура 4.11: Едностъпков преход в недетерминирана машина на Тюринг

Тази дефиниция на релацията $\vdash^\ell$ вече се повтаря няколко пъти.

Сега за всяко естествено число ℓ, ще дефинираме релацията $\vdash^\ell$, която ще казва, че от конфигурацията κ можем да достигнем до конфигурацията κ' за ℓ на брой стъпки.

$$\frac{}{\kappa \vdash^0 \kappa}\ \text{(рефлексивност)} \qquad \frac{\kappa \vdash \kappa'' \qquad \kappa'' \vdash^\ell \kappa'}{\kappa \vdash^{\ell+1} \kappa'}\ \text{(транзитивност)}$$

С $\vdash^\star$ ще означаваме рефлексивното и транзитивно затваряне на релацията $\vdash$ или с други думи,

$$\kappa \vdash^\star \kappa' \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[\kappa \vdash^\ell \kappa'].$$

Макар и една конфигурация κ да преставлява тройка, то често ще бъде удобно да гледаме на κ като на дума от $\Gamma^\star Q \Gamma^+$.

С $\vdash^\star$ ще означаваме рефлексивното и транзитивно затваряне на $\vdash$. Тогава за недетерминирана машина на Тюринг $\mathcal{N}$,

Дали да има $\sqcup$ за край на думата ?

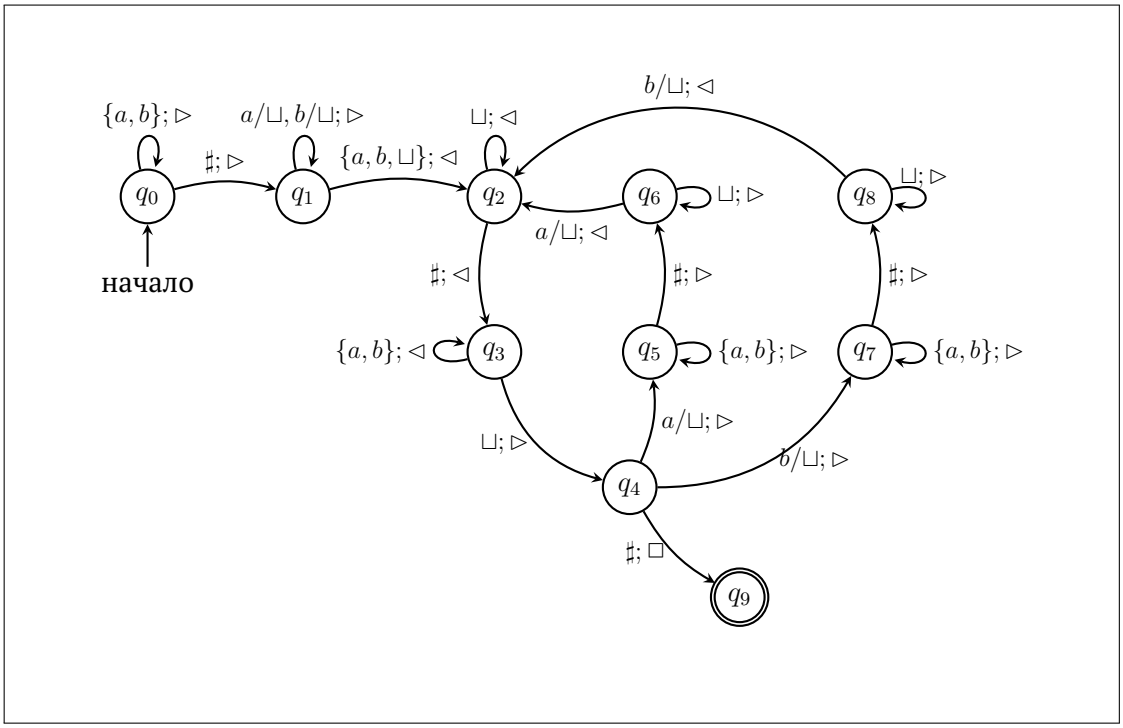
$$\mathcal{L}(\mathcal{N}) = \{\, \omega \in \Sigma^\star \mid (\varepsilon, q_{\texttt{start}}, \omega\sqcup) \vdash^\star_{\mathcal{N}} (\lambda, q_{\texttt{accept}}, \rho), \text{ за някои } \lambda, \rho \in \Gamma^\star \}.$$

Забележка. Върху дадена дума ω, недетерминираната машина на Тюринг $\mathcal{N}$ може да има много различни изчисления. Думата ω принадлежи на $\mathcal{L}(\mathcal{N})$ ако съществува *поне едно* изчисление, което завършва в състоянието $q_{\texttt{accept}}$. Възможно е много други изчисления при вход ω да завършват в $q_{\texttt{reject}}$ или никога да не завършват.

Аналогично, дефинираме една недетерминираната машина на Тюринг $\mathcal{N}$ да бъде **разрешител**, ако за всяка дума ω и *всяко изчисление* на $\mathcal{N}$ върху ω завършва в $q_{\texttt{accept}}$ или $q_{\texttt{reject}}$.

Пример 4.7. Нека да видим, че $L = \{\alpha\sharp\beta \mid \alpha, \beta \in \{a,b\}^\star \ \&\ \alpha \text{ е подниз на } \beta\}$ е разрешим език като построим недетерминираната машина на Тюринг $\mathcal{N}$, която разрешава този език.

Не е обяснено защо е разрешим.



Да видим, че $\mathcal{M}$ успешно разпознава думата $ab\sharp aabb$, която принадлежи на L.

$$
\begin{aligned}
(q_0, \underline{a}b\sharp aabb\sqcup) &\vdash (q_0, a\underline{b}\sharp aabb\sqcup) \vdash (q_0, ab\underline{\sharp}aabb\sqcup) \vdash (q_1, ab\sharp\underline{a}abb\sqcup) \\
&\vdash (q_1, ab\sharp \sqcup \underline{a}bb\sqcup) \vdash (q_2, ab\sharp\underline{\sqcup}abb\sqcup) \vdash (q_2, ab\underline{\sharp} \sqcup abb\sqcup) \\
&\vdash (q_3, a\underline{b}\sharp \sqcup abb\sqcup) \vdash (q_3, \underline{a}b\sharp \sqcup abb\sqcup) \vdash (q_3, \underline{\sqcup}ab\sharp \sqcup abb\sqcup) \\
&\vdash (q_4, \underline{a}b\sharp \sqcup abb\sqcup) \vdash (q_5, \sqcup\underline{b}\sharp \sqcup abb\sqcup) \vdash (q_5, \sqcup b\underline{\sharp} \sqcup abb\sqcup) \\
&\vdash (q_6, \sqcup b\sharp\underline{\sqcup}abb\sqcup) \vdash (q_6, \sqcup b\sharp \sqcup \underline{a}bb\sqcup) \vdash (q_2, \sqcup b\sharp\underline{\sqcup} \sqcup bb\sqcup) \\
&\vdash (q_2, \sqcup b\underline{\sharp} \sqcup \sqcup bb\sqcup) \vdash (q_3, \sqcup\underline{b}\sharp \sqcup \sqcup bb\sqcup) \vdash (q_3, \underline{\sqcup}b\sharp \sqcup \sqcup bb\sqcup) \\
&\vdash (q_4, \sqcup\underline{b}\sharp \sqcup \sqcup bb\sqcup) \vdash (q_7, \sqcup \sqcup \underline{\sharp} \sqcup \sqcup bb\sqcup) \\
&\vdash (q_8, \sqcup \sqcup \sharp\underline{\sqcup} \sqcup bb\sqcup) \vdash (q_8, \sqcup \sqcup \sharp \sqcup \underline{\sqcup}bb\sqcup) \\
&\vdash (q_8, \sqcup \sqcup \sharp \sqcup \sqcup\underline{b}b\sqcup) \vdash (q_2, \sqcup \sqcup \sharp \sqcup \underline{\sqcup} \sqcup b\sqcup) \\
&\vdash \cdots \vdash (q_4, \sqcup \sqcup \underline{\sharp} \sqcup \sqcup \sqcup b\sqcup) \vdash (q_9, \sqcup \sqcup \underline{\sharp} \sqcup \sqcup \sqcup b\sqcup)
\end{aligned}
$$

Канонична наредба на $\Sigma^\star$

За доказателството, че всяка НМТ е еквивалентна на ДМТ, е необходимо да фиксираме канонична подредба на думите над дадена азбука

Нека $\Sigma = \{a_0, a_1, \ldots, a_{k-1}\}$. Подреждаме думите по ред на тяхната дължина. Думите с еднаква дължина подреждаме по техния числов ред, т.е. гледаме на буквите a_i като числото i в k-ична бройна система. Тогава думите с дължина n са числата от 0 до $k^n - 1$ записани в k-ична бройна система. Ще означаваме с ω_i i-тата дума в $\Sigma^\star$ при тази подредба.

Ако $\Sigma = \{0, 1\}$, то наредбата започва така:

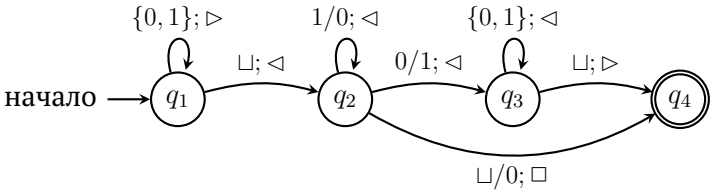
$$\varepsilon, 0, 1, \underbrace{00, 01, 10, 11}_{\text{от 0 до 3}}, \underbrace{000, 001, 010, 011, 100, 101, 110, 111}_{\text{от 0 до 7}}, 0000, 0001, \ldots$$

В този случай, $\omega_0 = \varepsilon$, $\omega_7 = 000$, $\omega_{13} = 110$. Обърнете внимание, че тази наредба отговаря на обхождане в широчина на едно пълно наредено двоично дърво. Можем да дефинираме и релацията $<_{\text{can}}$ по следния начин:

$$\alpha <_{\text{can}} \beta \overset{\text{деф}}{\Leftrightarrow} |\alpha| < |\beta| \vee (|\alpha| = |\beta| \,\&\, \alpha <_{\text{lex}} \beta).$$

Задача 4.4. Нека $\Sigma = \{a_0, \ldots, a_{k-1}\}$. Да разгледаме функцията $f : \Sigma^\star \to \Sigma^\star$, за която $f(\alpha)$ е думата веднага след α в каноничната подредба на $\Sigma^\star$. Докажете, че f е изчислима с еднолентова детерминираната машина на Тюринг.

Упътване. Ако $\Sigma = \{0, 1\}$, то машината на Тюринг има следния вид:



Фигура 4.13: Генериране на следващата дума в каноничната наредба.

□

Теорема 4.1. Ако L се разпознава от *недетерминирана* машина на Тюринг $\mathcal{N}$, то L е разпознава и от *детерминирана* машина на Тюринг $\mathcal{D}$.

В [HU79, стр. 164] не е добре обяснено.

Доказателство. Нека имаме недетерминирана машина на Тюринг $\mathcal{N}$, за която $L = \mathcal{L}(\mathcal{N})$. Една дума α принадлежи на $\mathcal{L}(\mathcal{N})$ точно тогава, когато съществува изчисление, което започва с думата α върху лентата и след краен брой стъпки, следвайки функцията на преходите $\Delta_{\mathcal{N}}$, достига до състоянието $q_{\texttt{accept}}$. Сложността идва от факта, че за думата α може да имаме много различни изчисления, като само някои от тях завършват в $q_{\texttt{accept}}$. Ще построим детерминирана машина на Тюринг $\mathcal{D}$,

която последователно ще симулира всички възможни *крайни* изчисления за думата α, докато намери такова, което завършва в състоянието $q_{\texttt{accept}}$.

На практика това, което правим е да представим всички възможни изчисления на $\mathcal{N}$ като r-разклонено дърво и да го обходим в широчина, докато не достигнем до $q_{\texttt{accept}}$

Лесно се съобразява, че всяко изчисление на $\mathcal{N}$ може да се представи като крайна редица от елементи на $Q \times \Gamma \times \{\triangleleft, \triangleright, \square\}$. Понеже това множество е крайно, то можем на всяка такава тройка да съпоставим естествено число $< r$, където

$$r = |Q| \cdot |\Gamma| \cdot 3.$$

Например, нека $Q = \{q_0, q_1\}$, $\Gamma = \{a, b\}$. Тогава можем да направим следната съпоставка:

$$\begin{aligned}
&(q_0, a, \square) \to 0,\ (q_0, a, \triangleleft) \to 1,\ (q_0, a, \triangleright) \to 2,\\
&(q_0, b, \square) \to 3,\ (q_0, b, \triangleleft) \to 4,\ (q_0, b, \triangleright) \to 5,\\
&(q_1, a, \square) \to 6, (q_1, a, \triangleleft) \to 7, (q_1, a, \triangleright) \to 8,\\
&(q_1, b, \square) \to 9, (q_1, b, \triangleleft) \to 10,\ (q_1, b, \triangleright) \to 11.
\end{aligned}$$

Ясно е, че всяко изчисление на $\mathcal{N}$ може да се представи като дума над азбуката $\Sigma = \{x_0, x_1, \ldots, x_{r-1}\}$. Например, изчислението от три стъпки

$$(\sqcup, q_0, aba) \vdash_N (b, q_1, ba) \vdash_{\mathcal{N}} (b, q_1, aa) \vdash_{\mathcal{N}} (ba, q_0, a)$$

може да се опише като думата $x_{11}x_6x_2$ над азбуката $\Sigma = \{x_0, x_1, \ldots, x_{11}\}$.

Детерминираната машина на Тюринг $\mathcal{D}$ има три ленти.

- На първата лента съхраняваме входящия низ и *тя никога не се променя*.
- На втората лента ще записваме последователно думи следвайки каноничната наредба на думите над азбуката $\{x_0, x_1, \ldots, x_{r-1}\}$. От *Задача 4.4* знаем как последователно да генерираме тези думи върху една лента.
- На третата лента симулираме изчислението на $\mathcal{N}$ върху думата от първата лента, използвайки изчислението, което е описано на втората лента. Например, ако съдържанието на втората лента е $x_{11}x_6x_2$, това означава, че симулираме изчисление от три стъпки като на първата стъпка избираме дванайсетата възможна тройка, на втората стъпка избираме седмата възможна тройка, на третата стъпка избираме третата възможна тройка.

  Ако симулацията завърши в състоянието $q_{\texttt{accept}}$ на $\mathcal{N}$, то машината $\mathcal{D}$ завършва успешно. В противен случай, на втората лента генерираме чрез функцията от *Задача 4.4* следващия низ относно каноничната наредба на $\{x_0, x_1 \ldots, x_{r-1}\}$; изтриваме третата лента, копираме първата лента на третата и започваме нова детерминистична симулация като думата върху втората лента ни ръководи какъв преход да правим на всяка стъпка.

□

Следствие 4.1. Ако L се разпознава от *недетерминиран* разрешител $\mathcal{N}$, то L също се разпознава от *детерминиран* разрешител $\mathcal{D}$.

Доказателство. Да разгледаме дървото T с крайно разклонение r, което представя всички изчисления на разрешителя $\mathcal{N}$ при вход думата ω. От *Лема 1.1* следва, че T е крайно дърво, да кажем с височина h, защото ако допуснем, че T е безкрайно, то ще има безкрайно дълго изчисление на $\mathcal{N}$, което е невъзможно, понеже $\mathcal{N}$ винаги достига до заключително състояние ($q_{\texttt{accept}}$ или $q_{\texttt{reject}}$).

- Ако $\mathcal{N}$ приема дадена дума ω, то детерминистичната ни симулация на $\mathcal{N}$ ще достигне до изчисление, кодирано като път в T, което завършва в състояние $q_{\texttt{accept}}$.
- Ако $\mathcal{N}$ не приема дадена дума ω, то детерминистичната ни симулация на $\mathcal{N}$ ще покаже, че всяко изчисление, кодирано като път в T, завършва в състояние $q_{\texttt{reject}}$. Един начин да направим това е да имаме една допълнителна лента, която използваме за брояч колко от възможните изчисления на $\mathcal{N}$ са завършили. Спираме, когато този брояч достигне r^h, където h е дължината на думата на втората лента, т.е. дълбочината на дървото на изчисленията на $\mathcal{N}$.

□

4.4 Основни свойства

Твърдение 4.3. Ако L е разрешим език над азбуката Σ, то $\Sigma^\star \setminus L$ също е разрешим език.

С други думи, разрешимите езици са затворени относно операцията допълнение. След малко в *Твърдение 4.6* ще видим, че това твърдение не е изпълнено за полуразрешими езици.

Упътване. Нека $L = \mathcal{L}(\mathcal{M})$, където $\mathcal{M}$ е разрешител. Нека $\mathcal{M}'$ е същата като $\mathcal{M}$, само със сменени $q_{\texttt{accept}}$ и $q_{\texttt{reject}}$ състояния. Тогава $\mathcal{M}'$ също е разрешител и $\overline{L} = \mathcal{L}(\mathcal{M}')$. □

Твърдение 4.4. Ако L_1 и L_2 са разрешими езици, то $L_1 \cup L_2$ е разрешим език.

С други думи, разрешимите езици са затворени относно операцията обединение. Като следствие получаваме, че всяко *крайно* обединение на разрешими езици е разрешим език.
✍ Съобразете, че това твърдение е изпълнено и за полуразрешими езици.

Упътване. Нека $L_1 = \mathcal{L}(\mathcal{M}_1)$ и $L_2 = \mathcal{L}(\mathcal{M}_2)$. Строим нова машина на Тюринг $\mathcal{M}$, която при вход думата α симулира едновременно изчисленията на $\mathcal{M}_1$ и $\mathcal{M}_2$ върху α. Това можем да направим като приемем, че $\mathcal{M}$ има две ленти - една за лентата на $\mathcal{M}_1$ и една за лентата на $\mathcal{M}_2$, като състоянията на $\mathcal{M}$ ще бъдат елементи на $Q_1 \times Q_2$. Ако една от двете машини достигне своето приемащо състояние, то $\mathcal{M}$ приема думата α. Ако и двете машини достигнат своите отхвърлящи състояния, то $\mathcal{M}$ отхвърля думата α. □

Твърдение 4.5. Ако L_1 и L_2 са разрешими езици, то $L_1 \cap L_2$ е разрешим език.

С други думи, разрешимите езици са затворени относно операцията сечение. Като следствие получаваме, че всяко *крайно* сечение на разрешими езици е разрешим език.
✍ Съобразете, че това твърдение е изпълнено и за полуразрешими езици.

Упътване. Нека $L_1 = \mathcal{L}(\mathcal{M}_1)$ и $L_2 = \mathcal{L}(\mathcal{M}_2)$. Строим нова машина на Тюринг $\mathcal{M}$, която при вход думата α симулира едновременно изчисленията на $\mathcal{M}_1$ и $\mathcal{M}_2$ върху α. Ако и двете машини достигнат до приемащите си състояния, то $\mathcal{M}$ приема думата α. Ако поне една от двете машини достигне до отхвърлящо състояние, то $\mathcal{M}$ отхвърля думата α. □

Теорема 4.2 (Клини-Пост). L и $\overline{L}$ са полуразрешими езици точно тогава, когато L е разрешим език.

Упътване. Посоката ($\Leftarrow$) е ясна. За посоката ($\Rightarrow$), нека $L = \mathcal{L}(\mathcal{M}_1)$ и $\overline{L} = \mathcal{L}(\mathcal{M}_2)$. Строим разрешител $\mathcal{M}$, която при вход думата α симулира едновременно изчисленията на $\mathcal{M}_1$ и $\mathcal{M}_2$ върху α. Например, може $\mathcal{M}$ да има две ленти за симулацията на $\mathcal{M}_1$ и $\mathcal{M}_2$. Знаем със сигурност, че точно едно от двете симулирани изчисления ще завърши в приемащо състояние. Ако това е $\mathcal{M}_1$, то $\mathcal{M}$ приема α. Ако това е $\mathcal{M}_2$, то $\mathcal{M}$ отхвърля α. □

Кодиране на машина на Тюринг

Да приемем, че:

Тук е удобно да индексираме от 1 вместо от 0.

- $Q = \{q_1, q_2, \ldots, q_n\}$, където $n \geq 2$;

- $q_1 = q_{\texttt{start}}$, $q_2 = q_{\texttt{accept}}$ и $q_3 = q_{\texttt{reject}}$.
- $\Sigma = \{x_1, \ldots, x_\ell\}$;
- $\Gamma = \{x_1, x_2, \ldots, x_s\}$, където $\ell < s$ и $x_s = \sqcup$;
- $D_1 = \square$, $D_2 = \triangleleft$, $D_3 = \triangleright$;

Аналогично така можем да кодираме и преходите на недетерминистична машина на Тюринг.

Така можем да кодираме преходите на детерминистична машина на Тюринг като думи. Да разгледаме прехода $\delta(q_i, x_j) = (q_k, x_t, D_m)$. Кодираме този преход със следната дума:

$$0^i 1 0^j 1 0^k 1 0^t 1 0^m.$$

Да обърнем внимание, че в този двоичен код няма последователни единици и той започва и завършва с нула.

За да кодираме една машина на Тюринг $\mathcal{M}$ е достатъчно да кодираме функцията на преходите δ. Понеже δ е крайна функция, нека с числото r да означим броя на всички възможни преходи. По описания по-горе начин, нека $\texttt{code}_i$ е числото в двоичен запис, получено за i-тия преход на δ. Тогава кодът на $\mathcal{M}$ е следното число в двоичен запис:

$$\ulcorner \mathcal{M} \urcorner \stackrel{\text{деф}}{=} 1110^\ell 11\ \texttt{code}_1\ 11\ \texttt{code}_2\ 11\ \cdots\ 11\ \texttt{code}_r\ 111.$$

Лесно се съобразява, че за две машини на Тюринг $\mathcal{M}$ и $\mathcal{M}'$ с различни функции на преходите, имаме $\ulcorner \mathcal{M} \urcorner \neq \ulcorner \mathcal{M}' \urcorner$. Ще означаваме с $\mathcal{M}_\omega$ машината на Тюринг, чийто код е ω, т.е. $\ulcorner \mathcal{M}_\omega \urcorner = \omega$.

По същия начин можем да дефинираме и код на краен автомат и стеков автомат.

Задача 4.5. Докажете, че езикът

$$L_{\texttt{code}} = \{\, \omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг} \,\}$$

е разрешим.

Диагоналният език

Теорема 4.3. Диагоналният език

$$L_{\texttt{diag}} \stackrel{\text{деф}}{=} \{\, \omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \omega \notin L(\mathcal{M}_\omega) \,\}$$

не се разпознава от машина на Тюринг, т.е. $L_{\texttt{diag}}$ **не** е полуразрешим език.

Това е версия на диагоналния метод на Кантор, с чиято помощ се доказва, че реалните числа са неизброимо много, т.е. има повече реални числа отколкото естествените.

Доказателство. Да допуснем, че $L_{\texttt{diag}}$ се разпознава от машина на Тюринг $\mathcal{D}$, т.е. $L_{\texttt{diag}} = \mathcal{L}(\mathcal{D})$. Тогава да видим какво имаме за думата $\ulcorner \mathcal{D} \urcorner$:

$$\ulcorner \mathcal{D} \urcorner \in L_{\texttt{diag}} \implies \ulcorner \mathcal{D} \urcorner \in \mathcal{L}(\mathcal{D}) \implies \ulcorner \mathcal{D} \urcorner \notin L_{\texttt{diag}},$$
$$\ulcorner \mathcal{D} \urcorner \notin L_{\texttt{diag}} \implies \ulcorner \mathcal{D} \urcorner \notin \mathcal{L}(\ulcorner \mathcal{D} \urcorner) \implies \ulcorner \mathcal{D} \urcorner \in L_{\texttt{diag}}.$$

Достигаме до противоречие. □

Тук е добре една безкрайна таблица да се нарисува.

Твърдение 4.6. Езикът

$$L_{\texttt{accept}} \stackrel{\text{деф}}{=} \{\, \omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \omega \in \mathcal{L}(\mathcal{M}_\omega) \,\}$$

е полуразрешим, но не е разрешим.

Упътване. Лесно се съобразява, че $L_{\texttt{accept}}$ е полуразрешим. Дефинираме (многолентова) машина на Тюринг $\mathcal{M}'$, която работи по следния начин:

- вход дума α;
- $\mathcal{M}'$ проверява дали α има вида $\ulcorner\mathcal{M}\urcorner$, за някоя машина на Тюринг $\mathcal{M}$;
- Ако α няма вида $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ завършва като отхвърля думата α.
- Ако $\alpha = \ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ симулира работата на $\mathcal{M}$ върху α. Тогава:
  - Ако $\mathcal{M}$ завърши след краен брой стъпки като приема α, то $\mathcal{M}'$ приема α.
  - Ако $\mathcal{M}$ завърши след краен брой стъпки като отхвърля α, то $\mathcal{M}'$ отхвърля α.
  - Ако $\mathcal{M}$ никога не завършва върху α, то $\mathcal{M}'$ също никога не завършва върху α.

Получаваме, че

$$\alpha \in L_{\texttt{accept}} \;\Leftrightarrow\; \alpha \in \mathcal{L}(\mathcal{M}'),$$

откъдето следва, че $L_{\texttt{accept}}$ е полуразрешим език.

Ако допуснем, че $L_{\texttt{accept}}$ е разрешим, то езикът $L_{\texttt{code}} \setminus L_{\texttt{accept}} = L_{\texttt{diag}}$ би бил разрешим, което е противоречие, защото $L_{\texttt{diag}}$ не е е дори полуразрешим. □

Следствие 4.2. Съществуват полуразрешим език L, за който $\overline{L}$ не е полуразрешим.

Задача 4.6. Докажете, че езикът

$$L_{\texttt{halt}} = \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{M}_\omega \text{ спира при вход } \omega\}$$

е полуразрешим, но не е разрешим.

Универсална машина на Тюринг

> A man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine. (Turing 1948: 416)

Можем за простота да считаме, че всички разглеждани машини на Тюринг са дефинирани над азбуката $\{0, 1\}$.

Теорема 4.4. Универсалният език

$$L_{\mathtt{univ}} \stackrel{\text{деф}}{=} \{\, \ulcorner \mathcal{M} \urcorner \sharp \omega \mid \mathcal{M} \text{ е машина на Тюринг и } \omega \in \mathcal{L}(\mathcal{M}) \,\}$$

е полуразрешим, но **не** е разрешим.

Разсъждението е много сходно с това защо $L_{\mathtt{accept}}$ полуразрешим. Ще наричаме $\mathcal{U}$ *универсална машина на Тюринг*.

Упътване. Първо да съобразим защо $L_{\mathtt{univ}}$ е полуразрешим език. Дефинираме (многолентова) машина на Тюринг $\mathcal{U}$, която работи по следния начин:

- вход дума α;
- $\mathcal{U}$ проверява дали α има вида $\ulcorner \mathcal{M} \urcorner \sharp \omega$, за някоя машина на Тюринг $\mathcal{M}$ и дума ω. Това става лесно, защото ω започва веднага след второ срещане на 111 в α.
- Ако α е от вида $\ulcorner \mathcal{M} \urcorner \sharp \omega$, то $\mathcal{U}$ симулира работата на $\mathcal{M}$ върху ω.
  - Ако $\mathcal{M}$ завърши след краен брой стъпки като приеме ω, то $\mathcal{U}$ приема α.
  - Ако $\mathcal{M}$ завърши след краен брой стъпки като отхвърли ω, то $\mathcal{U}$ отхвърля α.
  - Ако $\mathcal{M}$ никога не завършва върху ω, то очевидно $\mathcal{U}$ също никога не завършва върху α.
- Ако α няма вида $\ulcorner \mathcal{M} \urcorner \sharp \omega$, то $\mathcal{U}$ завършва веднага като отхвърля думата α.

Получаваме, че

$$\alpha \in L_{\mathtt{univ}} \;\Leftrightarrow\; \alpha \in \mathcal{L}(\mathcal{U}).$$

Сега да съобразим защо $L_{\mathtt{univ}}$ не е разрешим език. За произволна дума ω имаме:

$$\omega \in L_{\mathtt{accept}} \;\Leftrightarrow\; \omega\sharp\omega \in L_{\mathtt{univ}}.$$

Ако допуснем, че $L_{\mathtt{univ}}$ е разрешим, то тогава $L_{\mathtt{accept}}$ е разрешим език, което е противоречие. □

Следствие 4.3. Езикът

$$L'_{\mathtt{univ}} \stackrel{\text{деф}}{=} \{\ulcorner \mathcal{M} \urcorner \sharp \omega \mid \ulcorner \mathcal{M} \urcorner \text{ е машина на Тюринг и } \omega \notin \mathcal{L}(\mathcal{M})\}$$

не е полуразрешим.

4.5 Критерий за разрешимост

Сипсър нарича $\leq_m$ *mapping reducibility* [Sip12, 235].

Доказателството, че $L_{\texttt{univ}}$ не е разрешим е пример за една обща схема, с която можем да докажем, че даден език не е разрешим:

- Нека имаме езика K, за който вече знаем, че не е разрешим. В нашия пример, $K = L_{\texttt{accept}}$.
- Питаме се дали някой друг език L е разрешим.
- Намираме изчислима тотална функция f, за която е изпълнено, че:

$$\omega \in K \;\Leftrightarrow\; f(\omega) \in L.$$

  В *Теорема 4.4*, това е функцията $f(\omega) = \omega\sharp\omega$.

- В този случай ще означаваме $K \leq_m L$.
- Тогава, ако L е разрешим ще следва, че K е разрешим, което е противоречие.

Сега искаме да разгледаме един критерий, който ще ни казва кога един език съставен от кодове на машини на Тюринг е разрешим. С негова помощ ще можем директно да решаваме наглед трудни задачи. Например, в момента не е очевидно защо следния език не е разрешим:

$$L_{\texttt{palin}}\{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}) \\ \text{съдържа само думи палиндроми}\}.$$

След малко ще видим, че според критерия, който ще разгледаме, директно ще можем да заключим, че $L_{\texttt{palin}}$ не е разрешим. Да започнем с няколко примера.

Твърдение 4.7. Докажете, че езикът

$$L_{\Sigma^\star} \overset{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

не е разрешим.

Доказателство. Ще покажем, че съществува тотална изчислима функция f, за която:

$L_{\Sigma^\star}$ не е дори полуразрешим, но за момента не знаем как да докажем това.

$$\alpha \in L_{\texttt{accept}} \;\Leftrightarrow\; f(\alpha) \in L_{\Sigma^\star}.$$

Ще опишем алгоритъм (формално машина на Тюринг), за който при входна думата ω прави следното:

- Ако ω не е код на машина на Тюринг, то връщаме ω.

За различни $\mathcal{M}$ получаваме различни $\mathcal{M}'$.

- Ако ω е код на машина на Тюринг $\mathcal{M}$, то тук става интересно. Връщаме код на друга машина на Тюринг $\mathcal{M}'$, която работи по следния начин:
  - Вход дума α;
  - Първоначално $\mathcal{M}'$ не обръща внимание на α.
  - $\mathcal{M}'$ симулира работата на $\mathcal{M}$ върху думата $\ulcorner\mathcal{M}\urcorner$;
    * Ако след краен брой стъпки $\mathcal{M}$ завърши като приеме думата $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ приема думата α, т.е. $\mathcal{M}'$ завършва в състоянието $q_{\texttt{accept}}$.
    * Ако след краен брой стъпки $\mathcal{M}$ завърши като отхърли думата $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ отхвърля думата α, т.е. $\mathcal{M}'$ завършва в състоянието $q_{\texttt{reject}}$.
    * В противен случай, $\mathcal{M}$ никога не завършва върху $\ulcorner\mathcal{M}\urcorner$. Това означава, че $\mathcal{M}'$ никога не завършва върху входа α и следователно $\mathcal{M}'$ не приема думата α.

Получаваме, че:

$$\ulcorner\mathcal{M}\urcorner \in L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}') = \Sigma^\star \implies \ulcorner\mathcal{M}'\urcorner \in L_{\Sigma^\star},$$
$$\ulcorner\mathcal{M}\urcorner \notin L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}') = \emptyset \implies \ulcorner\mathcal{M}'\urcorner \notin L_{\Sigma^\star}.$$

На практика гореописаният алгоритъм дефинира тоталната изчислима функция

$$f(\omega) = \begin{cases} \ulcorner\mathcal{M}'\urcorner, & \text{ако } \omega = \ulcorner\mathcal{M}\urcorner \\ \omega, & \text{иначе.} \end{cases}$$

Тогава получаваме, че

$$\omega \in L_{\texttt{accept}} \Leftrightarrow f(\omega) \in L_{\Sigma^\star}$$

и ако допуснем, че $L_{\Sigma^\star}$ е разрешим език, то $L_{\texttt{accept}}$ също ще е разрешим, което е противоречие. □

Следствие 4.4. Езикът

$$\overline{L}_\emptyset \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \neq \emptyset\}$$

е полуразрешим, но не е разрешим.

Следствие 4.5. Езикът

$$L_\emptyset \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

не е полуразрешим.

Упътване. Ако $L_\emptyset$ беше разрешим, то неговото допълнение

$$\overline{L}_\emptyset = L_{\texttt{code}} \setminus L_{\texttt{Empty}}$$

щеше да е разрешим език, което е противоречие.

Ако $L_\emptyset$ беше полуразрешим, тогава, използвайки, че $\overline{L}_\emptyset$ е полуразрешим, от теоремата на Клини-Пост щеше да следва, че $L_\emptyset$ е разрешим, което е противоречие □

Задача 4.7. Докажете, че езикът

$$L_{\texttt{Dec}} = \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг, която е разрешител }\}$$

не е разрешим.

Твърдение 4.8. Езикът

$$L_{\texttt{reg}} \stackrel{\text{деф}}{=} \{\, \omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е регулярен език }\}$$

не е разрешим.

Доказателство. Да фиксираме един език, за който знаем, че не е регулярен, например, $\{0^n1^n \mid n \in \mathbb{N}\}$. Дефинираме алгоритъм, за който по вход $\ulcorner\mathcal{M}\urcorner$ връща код на машината на Тюринг $\mathcal{M}'$, която работи по следния начин: [Sip12, стр. 219]

- Вход думата α;
- Ако $\alpha = 0^n1^n$, за някое n, то $\mathcal{M}'$ приема думата α.
- Ако α не е от вида 0^n1^n, тогава $\mathcal{M}'$ симулира $\mathcal{M}$ върху думата $\ulcorner\mathcal{M}\urcorner$.
  - Ако след краен брой стъпки $\mathcal{M}$ завърши като приеме думата $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ приема α.
  - Ако след краен брой стъпки $\mathcal{M}$ завърши като отхвърли думата $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ отхвърля думата α.
  - В противен случай, $\mathcal{M}$ никога не завършва върху $\ulcorner\mathcal{M}\urcorner$. Това означава, че $\mathcal{M}'$ никога не завършва върху входа α и следователно $\mathcal{M}'$ не приема думата α.

Получаваме, че:

Използваме наготово, че $\Sigma^\star$ е регулярен език.

$$\begin{aligned}
\ulcorner\mathcal{M}\urcorner \in L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}') = \Sigma^\star \implies \ulcorner\mathcal{M}'\urcorner \in L_{\texttt{reg}},\\
\ulcorner\mathcal{M}\urcorner \notin L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}') = \{0^n1^n \mid n \in \mathbb{N}\} \implies \ulcorner\mathcal{M}'\urcorner \notin L_{\texttt{reg}}.
\end{aligned}$$

Сега вече трябва да е ясно, че следната тотална функция е изчислима:

$$f(\omega) = \begin{cases} \ulcorner\mathcal{M}'\urcorner, & \text{ако } \omega = \ulcorner\mathcal{M}\urcorner \\ \omega, & \text{иначе.} \end{cases}$$

Тогава получаваме, че

$$\omega \in L_{\texttt{accept}} \;\Leftrightarrow\; f(\omega) \in L_{\texttt{reg}}$$

и ако допуснем, че $L_{\texttt{reg}}$ е разрешим език, то $L_{\texttt{accept}}$ също ще е разрешим, което е противоречие. □

Сега ще видим, че идеята, която следвахме в горните доказателства може да се обобщи. Нека $\mathscr{S}$ е множество от полуразрешими езици над фиксирана азбука Σ. Например,

$$\mathscr{S} = \{L \subseteq \Sigma^\star \mid L \text{ е регулярен език}\}.$$

Ще казваме, че $\mathscr{S}$ е свойство на полуразрешимите езици. $\mathscr{S}$ е **тривиално свойство**, ако $\mathscr{S} = \emptyset$ или $\mathscr{S}$ съдържа точно всички полуразрешими езици. Нека разгледаме изброимото множество от всички машини на Тюринг, които разпознават езиците от $\mathscr{S}$. Ще представим това множество като език от кодовете на тези машини на Тюринг, т.е.

$$\texttt{Code}(\mathscr{S}) \stackrel{\text{деф}}{=} \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \in \mathscr{S}\}.$$

Можем да дефинираме и $\texttt{Code}(L)$, което е безкрайно изброимо множество, ако L е полуразрешим език.

[HU79, стр. 188]

Теорема 4.5 (Райс [Ric53])**.** За всяко нетривиално свойство $\mathscr{S}$ на полуразрешимите езици, $\texttt{Code}(\mathscr{S})$ е неразрешим.

Цел: да сведем ефективно $L_{\texttt{accept}}$ към $L_{\mathscr{S}}$

Доказателство. Без ограничение на общността, нека $\emptyset \notin \mathscr{S}$. Понеже $\mathscr{S}$ е нетривиално свойство, да разгледаме езика $L \in \mathscr{S}$, като $\mathcal{M}_L$ е машина на Тюринг, за която $\mathcal{L}(\mathcal{M}_L) = L$. Да разгледаме алгоритъм, който по дадена дума $\ulcorner\mathcal{M}\urcorner$ връща код на машина на Тюринг $\mathcal{M}'$, която зависи от $\mathcal{M}$ и от $\mathcal{M}_L$. Тя работи по следния начин:

Неформално описваме функцията δ за $\mathcal{M}'$

- вход думата α;
- първоначално $\mathcal{M}'$ не обръща внимание на α;
- $\mathcal{M}'$ симулира $\mathcal{M}$ върху думата $\ulcorner\mathcal{M}\urcorner$.
  - ако след краен брой стъпки $\mathcal{M}$ завърши като приеме думата $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ симулира $\mathcal{M}_L$ върху входната дума α;

    в този случай ще получим, че $\mathcal{L}(\mathcal{M}') = L$

    * ако след краен брой стъпки $\mathcal{M}_L$ завърши като приеме думата α, то $\mathcal{M}'$ приема α;
    * ако след краен брой стъпки $\mathcal{M}_L$ завърши като отхвърли думата α, то $\mathcal{M}'$ отхвърля α;
    * ако $\mathcal{M}_L$ никога не завършва върху α, то $\mathcal{M}'$ никога няма да завърши върху α и следователно $\mathcal{M}'$ не приема α.
  - ако след краен брой стъпки $\mathcal{M}$ завърши като отхвърли думата $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ отхвърля α;

    при тези два случая ще получим, че $\mathcal{L}(\mathcal{M}') = \emptyset$
  - Ако $\mathcal{M}$ никога не свършва върху $\ulcorner\mathcal{M}\urcorner$, то $\mathcal{M}'$ никога няма да свърши върху α, което означава, че $\mathcal{M}'$ не приема α.

От всичко това следва, че така описаната машина на Тюринг $\mathcal{M}'$ има свойствата:

$$\ulcorner\mathcal{M}\urcorner \in L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}') = L \implies \mathcal{L}(\mathcal{M}') \in \mathscr{S},$$
$$\ulcorner\mathcal{M}\urcorner \notin L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}') = \emptyset \implies \mathcal{L}(\mathcal{M}') \notin \mathscr{S}.$$

Сега вече трябва да е ясно, че гореописаният алгоритъм дефинира тоталната изчислима функция

$$f(\omega) = \begin{cases} \ulcorner \mathcal{M}' \urcorner, & \text{ако } \omega = \ulcorner \mathcal{M} \urcorner \\ \omega, & \text{иначе.} \end{cases}$$

Тогава получаваме, че

$$\omega \in L_{\texttt{accept}} \;\Leftrightarrow\; f(\omega) \in \texttt{Code}(\mathscr{S})$$

и ако допуснем, че $\texttt{Code}(\mathscr{S})$ е разрешимо множество, то ще следва, че $L_{\texttt{accept}}$ е разрешимо, което е противоречие.

Ако $\emptyset \in \mathscr{S}$, то правим горните разсъждения за класа от езици

$$\overline{\mathscr{S}} = \{ L \subseteq \Sigma^\star \mid L \text{ е полуразрешим език и } L \notin \mathscr{S} \}.$$

По аналогичен начин доказваме, че $\texttt{Code}(\overline{\mathscr{S}})$ не е разрешим език. Понеже

$$\texttt{Code}(\overline{\mathscr{S}}) = L_{\texttt{code}} \setminus \texttt{Code}(\mathscr{S}),$$

то $\texttt{Code}(\mathscr{S})$ също не е разрешим език. □

Следствие 4.6. За всяко от следните свойства $\mathscr{S}$ на полуразрешимите езици, $\texttt{Code}(\mathscr{S})$ **не** е разрешим език, където:

Тук няма нужда нищо да доказваме. Просто съобразяваме, че всяко от тези свойства на полуразрешимите езици е нетривиално.

а) $\mathscr{S}$ е свойството празнота, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) = \emptyset\}$$

не е разрешим;

б) $\mathscr{S}$ е свойството за пълнота, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) = \Sigma^\star\}$$

не е разрешим;

в) $\mathscr{S}$ е свойството крайност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } |\mathcal{L}(\mathcal{M})| < \infty\}$$

не е разрешим;

г) $\mathscr{S}$ е свойството безкрайност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } |\mathcal{L}(\mathcal{M})| = \infty\}$$

не е разрешим;

д) $\mathscr{S}$ е свойството регулярност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ е регулярен език}\}$$

не е разрешим;

Това свойство е нетривиално, защото вече показахме, че $\{a^n b^n c^n \mid n \in \mathbb{N}\}$ е полуразрешим (дори разрешим) език, а знаем отдавна, че този език не е безконтекстен.

е) $\mathscr{S}$ е свойството безконтекстност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ е безконтекстен}\}$$

не е разрешим;

Тук също - вече сме разгледали примери за полуразрешими езици, които не са разрешими.

ж) $\mathscr{S}$ е свойството разрешимост, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ е разрешим}\}$$

не е разрешим.

4.6 Критерии за полуразрешимост

Вече знаем, че на практика всички интересни въпроси за машини на Тюринг не са разрешими. Сега да видим какво можем да кажем за позитивната част на въпросите за машини на Тюринг.

Лема 4.1. Нека $\mathscr{S}$ е свойство на полуразрешимите езици. Ако съществува безкраен език $L_0 \in \mathscr{S}$, който няма крайно подмножество в $\mathscr{S}$, то $\texttt{Code}(\mathscr{S})$ *не е* полуразрешим език.

Това означава, че ако $\texttt{Code}(\mathscr{S})$ е полуразрешим език, то всеки език $L_0 \in \mathscr{S}$ притежава краен подезик, който също принадлежи на $\mathscr{S}$.

Упътване. Нека $L_0 = \mathcal{L}(\mathcal{M}_0)$ като $L_0 \in \mathscr{S}$, но всеки краен подезик на L_0 не принадлежи на $\mathscr{S}$. Ще докажем, че:

$$L_{\texttt{diag}} \leq_m \texttt{Code}(\mathscr{S}).$$

Ще дефинираме тотална изчислима функция f, която при вход думата $\omega \in \{0,1\}^\star$ работи по следния начин:

- Ако ω не е код на машина на Тюринг, то $f(\omega) \stackrel{\text{деф}}{=} \omega$.
- Ако ω е код на машината на Тюринг $\mathcal{M}_\omega$, то тогава $f(\omega) \stackrel{\text{деф}}{=} \ulcorner \mathcal{M}' \urcorner$, където $\mathcal{M}'$ работи така:
  - вход думата α;
  - $\mathcal{M}'$ симулира работата на $\mathcal{M}_\omega$ върху думата ω:
    * ако $\mathcal{M}_\omega$ завърши за по-малко от $|\alpha|$ на брой стъпки като *приеме* ω, то $\mathcal{M}'$ завършва веднага като *отхвърля* α;
    * в противен случай, $\mathcal{M}'$ симулира работата на $\mathcal{M}_0$ върху α.

  Така получаваме, че

$$\mathcal{L}(\mathcal{M}') = \begin{cases} \{\alpha \in L_0 \mid |\alpha| < s_0\}, & \text{ако } \mathcal{M}_\omega \text{ приема } \omega \\ L_0, & \text{ако } \mathcal{M}_\omega \text{ не приема } \omega, \end{cases}$$

  където s_0 е минималният брой стъпки необходими на $\mathcal{M}_\omega$ за да приеме думата ω.

  Заключаваме, че

$$\mathcal{M}_\omega \text{ не приема } \omega \implies \omega \in L_{\texttt{diag}} \implies \ulcorner \mathcal{M}' \urcorner \in \texttt{Code}(\mathscr{S})$$
$$\mathcal{M}_\omega \text{ приема } \omega \implies \omega \notin L_{\texttt{diag}} \implies \ulcorner \mathcal{M}' \urcorner \notin \texttt{Code}(\mathscr{S}).$$

Понеже

$$\omega \in L_{\texttt{diag}} \Leftrightarrow f(\omega) \in \texttt{Code}(\mathscr{S}),$$

то $\texttt{Code}(\mathscr{S})$ не е полуразрешим, защото от *Теорема 4.3* ние знаем, че $L_{\texttt{diag}}$ не е полуразрешим. □

Следствие 4.7. За всяко от следните свойства $\mathscr{S}$ на полуразрешимите езици, $\texttt{Code}(\mathscr{S})$ **не** е полуразрешим език, където:

Защо не можем да използваме *Лема 4.1* за да докажем, че свойството празнота не е полуразрешимо, както и свойството регулярност, разрешимост, полуразрешимост?

- $\mathscr{S}$ е свойството безкрайност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } |\mathcal{L}(\mathcal{M})| = \infty\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството за пълнота, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) = \Sigma^\star\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството неразрешимост, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ не е разрешим}\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството неполуразрешимост, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ не е полуразрешим}\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството нерегулярност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ не е регулярен}\}$$

  не е полуразрешим.

Това означава, че за полуразрешим език $\texttt{Code}(\mathscr{S})$, ако $L_1 \in \mathscr{S}$ и $L_1 \subseteq L_2$, като L_2 е полуразрешим, то $L_2 \in \mathscr{S}$.

Лема 4.2. Нека L_1 е език в $\mathscr{S}$ и нека L_2 е полуразрешим език, като $L_1 \subset L_2$ и $L_2 \notin \mathscr{S}$. Тогава $\texttt{Code}(\mathscr{S})$ не е полуразрешим език.

Упътване. Нека $L_1 = \mathcal{L}(\mathcal{M}_1)$ и $L_2 = \mathcal{L}(\mathcal{M}_2)$. Ще докажем, че

$$L_{\texttt{diag}} \leq_m \texttt{Code}(\mathscr{S}).$$

Ще дефинираме тотална изчислима функция f, която при вход думата $\omega \in \{0,1\}^\star$ работи по следния начин:

- Ако ω не е код на машина на Тюринг, то $f(\omega) = \omega$.
- Ако ω е код на машината на Тюринг $\mathcal{M}_\omega$, тогава $f(\omega)$ ще бъде кода на машината на Тюринг $\hat{\mathcal{M}}$, която работи по следния начин:
  - вход думата α;
  - $\hat{\mathcal{M}}$ симулира едновременно две изчисления - $\mathcal{M}_1$ върху α и $\mathcal{M}_\omega$ върху ω докато намери стъпка s, такава че:

* ако $\mathcal{M}_1$ завършва за s на брой стъпки като приема думата α, то $\hat{\mathcal{M}}$ завършва като приема думата α;
* ако $\mathcal{M}_\omega$ завършва за s на брой стъпки като приема думата ω, то $\hat{\mathcal{M}}$ симулира работата $\mathcal{M}_2$ върху α.
* ако $\hat{\mathcal{M}}$ не намери такава стъпка, то е ясно, че $\hat{\mathcal{M}}$ никога не завършва върху α.

Съобразете сами, че получаваме следното:

$$\mathcal{L}(\hat{\mathcal{M}}) = \begin{cases} L_2, & \text{ако } \mathcal{M}_\omega \text{ приема } \omega \\ L_1, & \text{ако } \mathcal{M}_\omega \text{ не приема } \omega. \end{cases}$$

Заключаваме, че:

$$\omega \in L_{\mathtt{diag}} \;\Leftrightarrow\; f(\omega) \in \mathtt{Code}(\mathscr{S}),$$

защото $L_2 \notin \mathscr{S}$, а $L_1 \in \mathscr{S}$. Това означава, че ефективно можем да сведем въпрос за принадлежност в $L_{\mathtt{diag}}$ към въпрос за принадлежност в $\mathtt{Code}(\mathscr{S})$. Следователно, ако $\mathtt{Code}(\mathscr{S})$ е полуразрешим език, то $L_{\mathtt{diag}}$ е полуразрешим език, което е противоречие. □

Следствие 4.8. За всяко от следните свойства $\mathscr{S}$ на полуразрешимите езици, $\mathtt{Code}(\mathscr{S})$ **не** е полуразрешим език, където:

- $\mathscr{S}$ е свойството крайност, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } |\mathcal{L}(\mathcal{M})| < \infty\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството празнота, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) = \emptyset\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството разрешимост, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ е разрешим}\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството безконтекстност, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ е безконтекстен}\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството регулярност, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\ulcorner \mathcal{M} \urcorner \mid \mathcal{M} \text{ е машина на Тюринг и } \mathcal{L}(\mathcal{M}) \text{ е регулярен}\}$$

  не е полуразрешим.

Теорема 4.6 (Райс-Шапиро)**.** Нека $\texttt{Code}(\mathscr{S})$ е полуразрешим език. Тогава е изпълнено, че:

$$L \in \mathscr{S} \;\Leftrightarrow\; (\exists L_0 \subseteq \Sigma^\star)[L_0 \text{ е краен и } L_0 \subseteq L \;\&\; L_0 \in \mathscr{S}].$$

Доказателство. Посоката $(\Rightarrow)$ следва от *Лема 4.1*, докато посоката $(\Leftarrow)$ следва от *Лема 4.2*. □

4.7 Проблемът за съответствието на Пост

Пример за проблема за съответствието на Пост се нарича всяка крайна редица от елементи на $\Sigma^\star \times \Sigma^\star$, които ние ще означаваме така:

На англ. *Post's correspondence problem* [HMU01, стр. 392], но по-добре е обяснено в [Sip12, стр. 227]. Тези двойки от думи се наричат домино.

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_n\\ \beta_n\end{bmatrix}.$$

Всяка една редица от този вид се нарича *пример* за PCP. Една непразна редица от индекси $i_1, i_2, \ldots, i_n$ се нарича *решение* на PCP примера, ако е изпълнено, че:

Ако $|\alpha_i| = |\beta_i|$ за всяко i, то задачата е тривиална.

$$\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_n} = \beta_{i_1}\beta_{i_2}\cdots\beta_{i_n}.$$

Задача 4.8. Намерете решение на следния пример за PCP:

[Sip12, стр. 239]

$$\begin{bmatrix}ab\\ abab\end{bmatrix}, \begin{bmatrix}b\\ a\end{bmatrix}, \begin{bmatrix}aba\\ b\end{bmatrix}, \begin{bmatrix}aa\\ a\end{bmatrix}.$$

Решение.

$$\begin{bmatrix}ab\cdot ab\cdot aba\cdot b\cdot b\cdot aa\cdot aa\\ abab\cdot abab\cdot b\cdot a\cdot a\cdot a\cdot a\end{bmatrix}$$

□

Модифициран проблем за съответствието

Казваме, че MPCP има решение, ако съществува произволна редица от индекси $i_1, \ldots, i_n$ (може и празна), такава че:

Тук искаме винаги да започваме с първото домино.

$$\alpha_1\alpha_{i_1}\cdots\alpha_{i_n} = \beta_1\beta_{i_1}\cdots\beta_{i_n}.$$

Лема 4.3. Съществува алгоритъм, който свежда MPCP към PCP.

Упътване. Нека имаме пример за MPCP:

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix}.$$

Нека символите $\star, \$$ не са от Σ. Нека за думата $\alpha = a_1\cdots a_n$ да дефинираме следните операции:

$$\begin{aligned}\star\,\alpha &= \star a_1 \star a_2 \cdots \star a_n\\ \alpha\star &= a_1 \star a_2 \star \cdots a_n\star\\ \star\,\alpha\star &= \star a_1 \star a_2 \star \cdots a_n \star.\end{aligned}$$

Тогава на базата на горния пример за MPCP, строим пример за PCP:

$$\begin{bmatrix}\star\alpha_1\star\\ \star\beta_1\end{bmatrix}, \begin{bmatrix}\alpha_1\star\\ \star\beta_1\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_k\star\\ \star\beta_k\end{bmatrix}, \begin{bmatrix}\$\\ \star\$\end{bmatrix}.$$

Така ние показахме, че

$$\texttt{MPCP} \leq_m \texttt{PCP}.$$

□

Следствие 4.9. Ако PCP е разрешим, то MPCP също е разрешим.

Ясно е, че проблемът на Пост е полуразрешим. Сега ще видим, че той не е разрешим.

Теорема 4.7 (Е. Пост [Pos46])**.** Проблемът за съответствието на Пост е неразрешим при азбука Σ с поне два символа.

Лесно се съобразява, че за азбука Σ само с една буква проблемът е разрешим.

Упътване. Нека приемем, че работим с машини на Тюринг, които не движат главата си наляво от левия край на лентата. Ще докажем, че $L_{\texttt{univ}} \leq_m \texttt{MPCP}$. Вече знаем, че MPCP се свежда алгоритмично към PCP , т.е. $\texttt{MPCP} \leq_m \texttt{PCP}$. Това означава, че ще опишем работата на тотална изчислима функция f, за която

$$\gamma \in L_{\texttt{univ}} \Leftrightarrow f(\gamma) \in \texttt{MPCP}.$$

Сега неформално ще опишем работата на функцията f. Нека фиксираме символа $\sharp \notin \Gamma$.

1) Нека имаме като вход дума $\gamma = \ulcorner \mathcal{M} \urcorner \sharp \alpha$.

Горната част на доминото се опитва да настигне долната част.

2) Започваме като добавяме за думата $\alpha = a_1 \cdots a_n$ над азбуката Σ следната двойка $\begin{bmatrix} \sharp \\ \sharp q a_1 \cdots a_n \sharp \end{bmatrix}$.

3) Ако $\delta(q, a) = (p, b, \triangleright)$, то добавяме двойката $\begin{bmatrix} qa \\ bp \end{bmatrix}$.

4) Ако $\delta(q, \sqcup) = (p, b, \triangleright)$, то добавяме и двойката $\begin{bmatrix} q\sharp \\ bp\sharp \end{bmatrix}$.

Тук е важно, че не позволяваме четящата глава да се мести по-наляво от първата клетка върху която е четящата глава при стартиране на изчислението.

5) Ако $\delta(q, a) = (p, b, \triangleleft)$, то добавяме двойките $\begin{bmatrix} xqa \\ pxb \end{bmatrix}$.

6) Ако $\delta(q, \sqcup) = (p, b, \triangleleft)$, то добавяме двойките $\begin{bmatrix} xq\sharp \\ pxb\sharp \end{bmatrix}$.

7) Ако $\delta(q, a) = (p, b, \square)$, то добавяме двойката $\begin{bmatrix} qa \\ pb \end{bmatrix}$.

8) за всеки $x \in \Gamma$, добавяме $\begin{bmatrix} x \\ x \end{bmatrix}$. Освен това, добавяме и двойката $\begin{bmatrix} \sharp \\ \sharp \end{bmatrix}$.

9) За всеки $x \in \Gamma$, добавяме двойката $\begin{bmatrix} xq_{\texttt{accept}} \\ q_{\texttt{accept}} \end{bmatrix}$ и $\begin{bmatrix} q_{\texttt{accept}}x \\ q_{\texttt{accept}} \end{bmatrix}$.

10) За да завършим, добавяме двойката $\begin{bmatrix} q_{\texttt{accept}}\#\# \\ \# \end{bmatrix}$.

Когато достигнем до приемащо състояние, то започваме да трием съдържанието на доминото за да можем да изравним двете части на доминото.

□

Следствие 4.10. Езикът

$$\texttt{AMBIG} = \{\ulcorner G \urcorner \mid G \text{ е неоднозначна безконтекстна граматика}\}$$

е неразрешим.

Ясно е, че AMBIG е полуразрешим език, нали?

Упътване. Да разгледаме един пример за PCP над азбуката Σ:

$$\begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix}, \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix}, \ldots, \begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix}.$$

По него можем ефективно да построим следната безконтекстна граматика:

$$\begin{aligned} S &\to A \mid B \\ A &\to \alpha_1 A c_1 \mid \alpha_2 A c_2 \mid \cdots \mid \alpha_n A c_n \mid \alpha_1 c_1 \mid \alpha_2 c_2 \mid \cdots \mid \alpha_n c_n \\ B &\to \beta_1 B c_1 \mid \beta_2 B c_2 \mid \cdots \mid \beta_n B c_n \mid \beta_1 c_1 \mid \beta_2 c_2 \mid \cdots \mid \beta_n c_n, \end{aligned}$$

където $c_1, \ldots, c_n \notin \Sigma$. Лесно се съобразява, че горният пример за PCP има решение точно тогава, когато безконтекстната граматика е неоднозначна. С други думи, показахме, че

$$\texttt{PCP} \leq_m \texttt{AMBIG}.$$

□

Следствие 4.11. Езикът

$$\texttt{INTERSECT} = \{\ulcorner G_1 \urcorner \# \ulcorner G_2 \urcorner \mid \mathcal{L}(G_1) \cap \mathcal{L}(G_2) \neq \emptyset\}$$

е неразрешим.

Ясно е, че INTERSECT е полуразрешим език, нали? За момента не е ясно дали допълнението на INTERSECT е полуразрешим език. Това ще разберем по-късно в *Теорема 4.8*.

Упътване. Да разгледаме един пример за PCP над азбуката Σ:

$$\begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix}, \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix}, \ldots, \begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix}.$$

По него можем ефективно да построим следните две безконтекстни граматика с правила:

$$\begin{aligned} S_1 &\to \alpha_1 S_1 c_1 \mid \alpha_2 S_1 c_2 \mid \cdots \mid \alpha_n S_1 c_n \mid \alpha_1 c_1 \mid \alpha_2 c_2 \mid \cdots \mid \alpha_n c_n \\ S_2 &\to \beta_1 S_2 c_1 \mid \beta_2 S_2 c_2 \mid \cdots \mid \beta_n S_2 c_n \mid \beta_1 c_1 \mid \beta_2 c_2 \mid \cdots \mid \beta_n c_n, \end{aligned}$$

където $c_1, \ldots, c_n \notin \Sigma$. Така показахме, че

$$\texttt{PCP} \leq_m \texttt{INTERSECT}.$$

□

4.8 Историята от всички изчисления

[HU79, стр. 201]

Да разгледаме машината на Тюринг $\mathcal{M}$. Една дума ω описва конфигурация на машина на Тюринг, ако $\omega \in \Gamma^\star Q \Gamma^\star$.

Твърдение 4.9. Да фиксираме една детерминистична машина на Тюринг $\mathcal{M}$. Тогава следните езици за безконтекстни:

$$\texttt{Valid}(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha \sharp \beta^{\mathtt{rev}} \mid \alpha, \beta \in \Gamma^\star Q \Gamma^\star \;\&\; \alpha \vdash_{\mathcal{M}} \beta \,\}$$
$$\texttt{Valid}'(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha^{\mathtt{rev}} \sharp \beta \mid \alpha, \beta \in \Gamma^\star Q \Gamma^\star \;\&\; \alpha \vdash_{\mathcal{M}} \beta \,\}$$
$$\texttt{Invalid}(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha \sharp \beta^{\mathtt{rev}} \mid \alpha, \beta \in \Gamma^\star Q \Gamma^\star \;\&\; \alpha \nvdash_{\mathcal{M}} \beta \,\}$$
$$\texttt{Invalid}'(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha^{\mathtt{rev}} \sharp \beta \mid \alpha, \beta \in \Gamma^\star Q \Gamma^\star \;\&\; \alpha \nvdash_{\mathcal{M}} \beta \,\}.$$

Упътване. Да напомним първо как дефинираме релацията $\vdash_{\mathcal{M}}$:

$$(\alpha_1, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1 y, p, \alpha_2) \qquad // \text{ ако } q \xrightarrow{x/y;\triangleright} p$$
$$(\alpha_1, q, \varepsilon) \vdash_{\mathcal{M}} (\alpha_1 y, p, \varepsilon) \qquad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleright} p$$
$$(\alpha_1 z, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1, p, zy\alpha_2) \qquad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$(\alpha_1 z, q, \varepsilon) \vdash_{\mathcal{M}} (\alpha_1, p, z) \qquad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleleft} p$$
$$(\varepsilon, q, x\alpha_2) \vdash_{\mathcal{M}} (\varepsilon, p, \sqcup y\alpha_2) \qquad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$(\alpha_1, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1, p, y\alpha_2) \qquad // \text{ ако } q \xrightarrow{x/y;\square} p.$$

Думите в езика $\texttt{Valid}(\mathcal{M})$ кодират релацията $\vdash_{\mathcal{M}}$. Това означава, че всяка дума на $\texttt{Valid}(\mathcal{M})$ има някое от следните представяния:

Последните два случая не са нужни, ако машината на Тюринг не може да чете по-наляво от най-лявата клетка на входната дума.

$$\alpha_1 q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} p y \alpha_1^{\mathtt{rev}} \qquad // \text{ ако } q \xrightarrow{x/y;\triangleright} p$$
$$\alpha_1 q \sharp p y \alpha_1^{\mathtt{rev}} \qquad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleright} p$$
$$\alpha_1 z q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} y z p \alpha_1^{\mathtt{rev}} \qquad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$\alpha_1 z q \sharp y z p \alpha_1^{\mathtt{rev}} \qquad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleleft} p$$
$$\alpha_1 q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} y p \alpha_1^{\mathtt{rev}} \qquad // \text{ ако } q \xrightarrow{x/y;\square} p$$
$$q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} y \sqcup p \qquad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$q \sharp y \sqcup p \qquad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleleft} p$$

Ще опишем неформално стеков автомат P за езика $\texttt{Valid}(\mathcal{M})$. Нека

$$Q^P \stackrel{\text{деф}}{=} \{r_q \mid q \in Q^{\mathcal{M}}\} \cup \{r, \hat{r}\}.$$

- Първо четем α_1 и я записваме в стека като $\alpha_1^{\mathtt{rev}}$. Това правим като дефинираме функцията на преходите като

$$(\forall a \in \Sigma)(z \in \Gamma)[\, \Delta_P(r, a, z) \stackrel{\text{деф}}{=} \{(r, az)\}\,].$$

- Правим това докато не срещнем някое $q \in Q^{\mathcal{M}}$. Тогава трябва да направим преход на $\mathcal{M}$. Тук трябва да внимаваме, защото за да направим преход, трябва да знаем състоянието q и да прочетем следващия символ. Един начин да разрешим този проблем е като запомним кое състояние сме прочели на машината на Тюринг в състоянията на стековия автомат:

$Q^{\mathcal{M}}$ са букви от азбуката на стековия автомат P.

$$(\forall q \in Q^{\mathcal{M}})(\forall z \in \Gamma)[\, \Delta_P(r, q, z) = \{(r_q, z)\}\,].$$

  – ако $q \xrightarrow{x/y;\triangleright} p$, то слагаме yp на върха на стека, т.е.

Стекът представлява $\alpha_1^{\mathtt{rev}}$. Ако $\alpha_1 = \varepsilon$, то z е символа за дъно на стека.

$$\Delta_P(r_q, x, z) \ni (\hat{r}, pyz).$$

  – Трябва да внимаваме, когато имаме $q\sharp$, защото това се интерпретира като четящата глава да е върху $\sqcup$. Това означава, че ако $q \xrightarrow{\sqcup/y;\triangleright} p$, то трябва да имаме и следното:

$$\Delta_P(r_q, \sharp, z) \ni (r_\sharp, pyz).$$

  – ако $q \xrightarrow{x/y;\triangleleft} p$, то ако z е върха на стека, заменяме z с pzy, т.е.

$$\Delta_P(r_q, x, z) \ni (\hat{r}, yzp).$$

  – Отново трябва да внимаваме, когато имаме $q\sharp$, защото това се интерпретира като четящата глава да е върху $\sqcup$. Това означава, че ако $q \xrightarrow{\sqcup/y;\triangleleft} p$, то трябва да имаме и следното:

$$\Delta_P(r_q, \sharp, z) \ni (r_\sharp, yzp).$$

  – ако $q \xrightarrow{x/y;\square} p$, то ако z е върха на стека, заменяме z с ypz, т.е.

$$\Delta_P(r_q, x, z) \ni (\hat{r}, ypz).$$

  – Ако позволяваме машината на Тюринг да се движи по-наляво от най-лявата клетка на лентата, върху която е записана входната дума, то трябва да разгледаме и случая, когато $\alpha_1 = \varepsilon$, т.е. z е символа за дъно на стека. Тогава:

    * Ако $q \xrightarrow{x/y;\triangleleft} p$, то

$$\Delta_P(r_q, x, z) \ni (\hat{r}, y \sqcup pz).$$

    * Ако $q \xrightarrow{\sqcup/y;\triangleleft} p$, то

$$\Delta_P(r_q, \sharp, z) \ni (r_\sharp, y \sqcup pz).$$

- Сега вече сме в състояние $\hat{r}$ и остава да прочетем α_2 и да я запишем в стека като $\alpha_2^{\texttt{rev}}$:
$$\Delta_P(\hat{r}, x, z) = \{(\hat{r}, xz)\}.$$

За да разпознаем $\texttt{Invalid}(\mathcal{M})$ трябва само да разменим условията за приемане и отхвърляне на думата.

- Разбираме кога сме свършили с α_2 когато стигнем до $\sharp$. Преминаваме в състояние $r_\sharp$ и започваме да четем думата след $\sharp$ и сравняваме с това, което имаме в стека.
  - Ако намерим разлика, то отхвърляме думата.
  - Ако достигнем до дъното на стека, то приемаме думата.

□

Забележка. Да обърнем внимание, че горната конструкция на стековия автомат P е **ефективна**, т.е. съществува алгоритъм, който при вход код на машина на Тюринг $\mathcal{M}$ връща като изход код на стеков автомат P за езика $\texttt{Valid}(\mathcal{M})$.

История на машина на Тюринг

Дума от вида $\omega_1\sharp\omega_2\sharp\omega_3\sharp\omega_4\sharp\cdots\omega_n\sharp$ се нарича **история на приемащо изчисление** на машината на Тюринг $\mathcal{M}$, ако

- $\omega_i \in \Gamma^\star Q \Gamma^\star$, т.е. ω_i описва моментна конфигурация и ω_i не започва и не завършва на $\sqcup$.
- $\omega_1 \in q_{\texttt{start}}\Sigma^\star$ описва начална конфигурация.
- $\omega_n \in \Gamma^\star \cdot \{q_{\texttt{accept}}\} \cdot \Gamma^\star$ описва приемаща конфигурация.
- За четно $i < n$, $\omega_i^{\texttt{rev}} \vdash_{\mathcal{M}} \omega_{i+1}$;
- За нечетно $i < n$, $\omega_i \vdash_{\mathcal{M}} \omega_{i+1}^{\texttt{rev}}$;

Езикът съставен от всички такива думи ще означаваме с $\texttt{History}(\mathcal{M})$.

[HU79, стр. 201] **Лема 4.4.** За всяка машина на Тюринг $\mathcal{M}$ е изпълнено, че:
$$\texttt{History}(\mathcal{M}) = L_1 \cap L_2,$$
където L_1 и L_2 са безконтекстни езици. Освен това, граматиките на L_1 и L_2 могат ефективно да бъдат построени по кода на $\mathcal{M}$.

Упътване. Разгледайте следните безконтекстни езици:
$$L_1 \stackrel{\text{деф}}{=} (\texttt{Valid}(\mathcal{M}) \cdot \{\sharp\})^\star \cdot (\{\varepsilon\} \cup \Gamma^\star \cdot \{q_{\texttt{accept}}\} \cdot \Gamma^\star \cdot \{\sharp\})$$
$$L_2 \stackrel{\text{деф}}{=} \{q_{\texttt{start}}\} \cdot \Sigma^\star \cdot \{\sharp\} \cdot (\texttt{Valid'}(\mathcal{M}) \cdot \{\sharp\})^\star \cdot (\{\varepsilon\} \cup \Gamma^\star \cdot \{q_{\texttt{accept}}\} \cdot \Gamma^\star \cdot \{\sharp\}),$$

□

Лема 4.5. Нека $\mathcal{M}$ е детерминистична машина на Тюринг, която за всеки вход прави поне две стъпки преди да спре. Тогава $\texttt{History}(\mathcal{M})$ е безконтекстен (регулярен) точно тогава, когато $\mathcal{L}(\mathcal{M})$ е краен.

Упътване. Ясно е, че ако $\mathcal{L}(\mathcal{M})$ е краен, то $\texttt{History}(\mathcal{M})$ е краен и следователно безконтекстен (регулярен).

Нека сега $\mathcal{L}(\mathcal{M})$ е безкраен. Тогава за всяко $p \geq 1$ има думи $\omega \in \mathcal{L}(\mathcal{M})$, за които $|\omega| \geq p$. Тогава може да приложите лемата за покачването за да докажете, че $\texttt{History}(\mathcal{M})$ не е безконтекстен (регулярен). □

Задача 4.9. Обяснете как може ефективно да се кодира всяка безконтекстна граматика G като дума $\ulcorner G \urcorner$ над азбуката $\{0, 1\}$.

Теорема 4.8. Езикът

$$\overline{\texttt{INTERSECT}} = \{\ulcorner G_1 \urcorner \cdot \ulcorner G_2 \urcorner \mid G_1 \text{ и } G_2 \text{ са безконт. грам. и } \mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \emptyset\}$$

не е полуразрешим.

Упътване. Лесно се вижда, че $L_{\texttt{Empty}} \leq_m \overline{\texttt{INTERSECT}}$. По дадена дума $\ulcorner \mathcal{M} \urcorner$, можем ефективно да намерим G_1 и G_2, за които $\mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \texttt{Accept}(\mathcal{M})$, т.е. съществува тотална изчислима функция f, за която

Да напомним, че от *Следствие 4.11* знаем, че INTERSECT не е разрешим, но е полуразрешим.

$$f(\ulcorner \mathcal{M} \urcorner) = \ulcorner G_1 \urcorner \cdot \ulcorner G_2 \urcorner.$$

Тогава ако L е полуразрешим език, то $L_{\texttt{Empty}}$ е полуразрешим език, което е противоречие, защото

$$\ulcorner \mathcal{M} \urcorner \in L_{\texttt{Empty}} \Leftrightarrow f(\ulcorner \mathcal{M} \urcorner) \in \overline{\texttt{INTERSECT}}.$$

□

Лема 4.6. За всяка машина на Тюринг $\mathcal{M}$, допълнението на $\texttt{History}(\mathcal{M})$, което означаваме като $\overline{\texttt{History}}(\mathcal{M})$, е безконтекстен език. Освен това, по кода на $\mathcal{M}$ можем ефективно да намерим код на безконтекстната граматика за $\overline{\texttt{History}}(\mathcal{M})$.

Упътване. Една дума α не е история на приемащо изчисление, ако е изпълнено някое от следните условия:

- α не е от вида $\omega_1 \sharp \omega_2 \sharp \cdots \sharp \omega_n \sharp$, където $\omega_i \in \Gamma^\star Q \Gamma^\star$, или
- ако α е от вида $\omega_1 \sharp \omega_2 \sharp \cdots \sharp \omega_n \sharp$, където $\omega_i \in \Gamma^\star Q \Gamma^\star$, тогава:
  - $\omega_1 \notin \{q_{\texttt{start}}\} \cdot \Gamma^\star$, или
  - $\omega_n \notin \Gamma^\star \cdot \{q_{\texttt{accept}}\} \cdot \Gamma^\star$, или

Можем да опишем това свойство с регулярен език

- $\omega_i \not\vdash_{\mathcal{M}} \omega_{i+1}^{\mathtt{rev}}$, т.е. $\omega_i \sharp \omega_{i+1}^{\mathtt{rev}} \in \mathtt{Invalid}(\mathcal{M})$, за някое нечетно $i < n$, или
- $\omega_i^{\mathtt{rev}} \not\vdash_{\mathcal{M}} \omega_{i+1}$, т.е. $\omega_i^{\mathtt{rev}} \sharp \omega_{i+1} \in \mathtt{Invalid'}(\mathcal{M})$, за някое четно $i < n$.

Думите притежаващи някое от тези свойства могат да се опишат като обединение на три регулярни езика, което е лесно защото регулярните езици са затворени относно допълнение, и двата безконтекстни езика $\mathtt{Invalid}(\mathcal{M})$ и $\mathtt{Invalid'}(\mathcal{M})$. □

Теорема 4.9. За дадена азбука Σ, езикът

$$\mathtt{All}_{\mathtt{CFG}} = \{\ulcorner G \urcorner \mid G \text{ е безконтекстна граматика и } \mathcal{L}(G) = \Sigma^\star\}$$

не е полуразрешим.

Упътване. Ще видим, че имаме следното свеждане:

$$L_{\mathtt{Empty}} \leq_m \mathtt{All}_{\mathtt{CFG}}.$$

Тук ще използваме, че ако $\mathcal{L}(\mathcal{M}) = \emptyset$, то $\overline{\mathtt{History}}(\mathcal{M}) = \hat{\Sigma}^\star$, където $\hat{\Sigma} = \Gamma \cup Q \cup \{\sharp\}$. По даден код на машината на Тюринг $\mathcal{M}$, можем ефективно да намерим код на безконтекстната граматика G, за която $\mathcal{L}(G)$ са точно невалидните изчисления на $\mathcal{M}$, т.е. съществува тотална изчислима f, за която

$$f(\ulcorner \mathcal{M} \urcorner) = \ulcorner G \urcorner \text{ и } \mathcal{L}(G) = \overline{\mathtt{History}}(\mathcal{M}).$$

Тогава, ако допуснем, че $\mathtt{All}_{\mathtt{CFG}}$ е полуразрешим език, то $L_{\mathtt{Empty}}$ е полуразрешим, защото

$$\ulcorner \mathcal{M} \urcorner \in L_{\mathtt{Empty}} \;\Leftrightarrow\; f(\ulcorner \mathcal{M} \urcorner) \in \mathtt{All}_{\mathtt{CFG}}.$$

□

Следствие 4.12. Следните езици не са полуразрешим:

а) $\{\ulcorner G_1 \urcorner \cdot \ulcorner G_2 \urcorner \mid G_1 \text{ и } G_2 \text{ са безконт. грам. и } \mathcal{L}(G_1) = \mathcal{L}(G_2)\}$;

б) $\{\ulcorner G_1 \urcorner \cdot \ulcorner G_2 \urcorner \mid G_1 \text{ и } G_2 \text{ са безконт. грам. и } \mathcal{L}(G_1) \subseteq \mathcal{L}(G_2)\}$;

в) $\{\ulcorner G \urcorner \cdot r \mid G \text{ е безконт. грам. и } r \text{ е рег. израз и } \mathcal{L}(G) = \mathcal{L}(r)\}$;

г) $\{\ulcorner G \urcorner \cdot \ulcorner \mathcal{A} \urcorner \mid G \text{ е безконт. грам. и } \mathcal{A} \text{ е ДКА и } \mathcal{L}(G) = \mathcal{L}(\mathcal{A})\}$;

д) $\{\ulcorner G \urcorner \cdot r \mid G \text{ е безконт. грам. и } r \text{ е рег. израз и } \mathcal{L}(r) \subseteq \mathcal{L}(G)\}$;

е) $\{\ulcorner G \urcorner \cdot \ulcorner \mathcal{A} \urcorner \mid G \text{ е безконт. грам. и } \mathcal{A} \text{ е ДКА и } \mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(G)\}$.

Забележка. Добре е да обърнем внимание, че езикът

$$L = \{\ulcorner G \urcorner \cdot \ulcorner \mathcal{A} \urcorner \mid G \text{ е безконт. грам. и } \mathcal{A} \text{ е ДКА и } \mathcal{L}(G) \subseteq \mathcal{L}(\mathcal{A})\}$$

е разрешим. Това е така, защото $\mathcal{L}(G) \subseteq \mathcal{L}(\mathcal{A}) \Leftrightarrow \mathcal{L}(G) \cap \mathcal{L}(\overline{\mathcal{A}}) = \emptyset$, защото сечението на безконтекстен и регулярен език е безконтекстен език.

Теорема 4.10 (Грейбах 1963)**.** Нека $\mathcal{C}$ е клас от езици, за който съществува ефективно кодиране $\ulcorner L \urcorner$ на езиците в $\mathcal{C}$ и който е:

[HU79, стр. 205]

По дадена дума ω можем ефективно да проверим дали тя кодира език от $\mathcal{C}$ или не.

- ефективно затворен относно обединение;
- ефективно затворен относно конкатенация с регулярен език;
- "$= \Sigma^\star$"е неразрешим за достатъчно голяма Σ.

Нека P е нетривиално свойство на $\mathcal{C}$, което е изпълнено за всеки регулярен език и ако $L \in P$, то $L/_a \in P$, където

Съществуват езици от $\mathcal{C}$, които не притежават свойството P и такива, които го притежават.

$$L/_a = \{\omega \mid \omega a \in L\}.$$

Тогава езикът $\{\ulcorner L \urcorner \mid P(L)\ \&\ L \in \mathcal{C}\}$ е неразрешим.

Упътване. Да фиксираме език $L_0 \in \mathscr{C}$, за който *не е изпълнено* свойството P. Нека да приемем, че $L_0 \subseteq \Sigma^\star$, която е достатъчно голяма азбука, за която въпроса "$= \Sigma^\star$" е неразрешим. За произволен език $L \in \mathcal{C}$, да разгледаме езика

$$\hat{L} \stackrel{\text{деф}}{=} L_0 \sharp \Sigma^\star \cup \Sigma^\star \sharp L.$$

Ясно е, че $\hat{L} \in \mathcal{C}$, защото $\mathcal{C}$ е ефективно затворен относно конкатенация с регулярен език и относно обединение. Първо ще докажем, че:

$$L = \Sigma^\star \Leftrightarrow \ulcorner \hat{L} \urcorner \in P. \tag{4.1}$$

- Ако $L = \Sigma^\star$, то $\hat{L}$ е регулярен, защото тогава $\hat{L} = \Sigma^\star \sharp \Sigma^\star$ е очевидно регулярен и от избора на P, $\ulcorner \hat{L} \urcorner \in \mathcal{C}$.
- Ако $L \neq \Sigma^\star$, то нека да фиксираме дума $\omega \notin L$. Ако допуснем, че $\ulcorner \hat{L} \urcorner \in P$, то езикът за езикът $\hat{L}/_{\sharp\omega} = L_0$ също ще е изпълнено свойството P, което е противоречие с избора на L_0.

Ако $\ulcorner L \urcorner \in P$, то за $L/_\beta \stackrel{\text{деф}}{=} \{\alpha \mid \alpha\beta \in L\}$ е изпълнено P.

От (4.1) следва, че P е разрешимо свойство точно тогава, когато въпросът "$= \Sigma^\star$" за езиците от $\mathcal{C}$ е разрешим, което е противоречие. □

Следствие 4.13. Въпросът дали една безконтекстна граматика описва регулярен език е неразрешим. По-точно, езикът

$$\texttt{Reg} = \{\ulcorner G \urcorner \mid G \text{ е безконтекстна граматика и } \mathcal{L}(G) \text{ е регулярен език}\}$$

е неразрешим.

Доказателство. Ясно е, че имаме ефективно кодиране на безконтекстните граматики $\ulcorner G \urcorner$ и освен това те са ефективно затворени относно конкатенация с регулярен език и относно обединение. Вече знаем от *Теорема 4.9*, че $= \Sigma^\star$ за безконтекстни граматики е неразрешим за достатъчно голяма азбука Σ. Тогава от теоремата на Грейбах следва, че Reg е неразрешим език. □

4.9 Сложност

- Казваме, че детерминистичната машина на Тюринг $\mathcal{M}$ е **полиномиално ограничена**, ако съществува полином $p(x)$, такъв че няма дума ω, за която $\mathcal{M}$ да извършва при вход ω повече от $p(|\omega|)$ стъпки.

- Езикът L се нарича **детерминистично полиномиално разрешим**, ако съществува полиномиално ограничен детерминистичен разрешител $\mathcal{M}$, за който $L = \mathcal{L}(\mathcal{M})$. Нека

$$\mathcal{P} \stackrel{\text{деф}}{=} \{L \subseteq \Sigma^\star \mid L \text{ е полиномиално разрешим с ДМТ}\}.$$

- Казваме, че детерминистичната машина на Тюринг $\mathcal{M}$ е **експоненциално ограничена**, ако съществува полином $p(x)$, такъв че няма дума ω, за която $\mathcal{M}$ да извършва при вход ω повече от $2^{p(|\omega|)}$ стъпки.

- Езикът L се нарича **детерминистично експоненциално разрешим**, ако съществува експоненциално ограничен детерминистичен разрешител $\mathcal{M}$, за който $L = \mathcal{L}(\mathcal{M})$. Нека

$$\mathcal{EXP} \stackrel{\text{деф}}{=} \{L \subseteq \Sigma^\star \mid L \text{ е експоненциално разрешим с ДМТ}\}.$$

- Казваме, че недетерминистичната машина на Тюринг $\mathcal{N}$ е **полиномиално ограничена**, ако съществува полином $p(x)$, такъв че няма дума ω, за която $\mathcal{N}$ да извършва при вход ω повече от $p(|\omega|)$ стъпки.

- Езикът L се нарича **недетерминистично полиномиално разрешим**, ако съществува полиномиално ограничена недетерминистичен разрешител $\mathcal{N}$, за който $L = \mathcal{L}(\mathcal{N})$. Нека

$$\mathcal{NP} \stackrel{\text{деф}}{=} \{L \subseteq \Sigma^\star \mid L \text{ е полиномиално разрешим с НМТ}\}.$$

Задача 4.10. Докажете, че класът $\mathcal{P}$ е затворен относно допълнение, обединение, сечение и конкатенация.

Задача 4.11. Докажете, че класът $\mathcal{P}$ е затворен относно операцията звезда на Клини.

Теорема 4.11. $\mathcal{NP} \subseteq \mathcal{EXP}$.

Твърдение 4.10. За азбука Σ от поне две букви, можем да обобщим някои от резултатите от предишните глави:

$$\texttt{REG} \subsetneqq \texttt{CFG} \subsetneqq \mathcal{P}.$$

Упътване. Езикът $\{a^n b^n c^n \mid n \in \mathbb{N}\} \in \mathcal{P}$, но не е безконтекстен. □

4.10 Задачи

Задача 4.12. Вярно ли е, че следните езици са разрешими?

а) $\{\ulcorner\mathcal{A}\urcorner \cdot \omega \mid \mathcal{A}$ е ДКА и $\omega \in \mathcal{L}(\mathcal{A})\}$;

б) $\{\ulcorner\mathcal{A}\urcorner \mid \mathcal{A}$ е ДКА и $\mathcal{L}(\mathcal{A})$ е безкраен език$\}$;

в) $\{\ulcorner\mathcal{A}\urcorner \mid \mathcal{A}$ е ДКА и $\mathcal{L}(\mathcal{A}) = \{0,1\}^\star\}$;

г) $\{\ulcorner\mathcal{A}\urcorner \mid \mathcal{A}$ е ДКА и $\mathcal{L}(\mathcal{A})$ съдържа поне една дума с равен брой нули и единици$\}$;

д) $\{\ulcorner\mathcal{A}\urcorner \mid \mathcal{A}$ е ДКА и $\mathcal{L}(\mathcal{A})$ съдържа поне една дума палиндром$\}$;

е) $\{\ulcorner\mathcal{A}\urcorner \mid \mathcal{A}$ е ДКА и $\mathcal{L}(\mathcal{A})$ не съдържа дума с нечетен брой единици$\}$;

ж) $\{\ulcorner\mathcal{A}\urcorner \cdot \ulcorner\mathcal{B}\urcorner \mid \mathcal{A}$ и $\mathcal{B}$ са ДКА и $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})\}$;

з) $\{\ulcorner\mathcal{A}\urcorner \cdot \ulcorner\mathcal{B}\urcorner \mid \mathcal{A}$ и $\mathcal{B}$ са ДКА и $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})\}$;

Задача 4.13. Вярно ли е, че следните езици са разрешими?

а) $\{\ulcorner G\urcorner \cdot \omega \mid G$ е безконтекстна граматика и $\omega \in \mathcal{L}(G)\}$;

б) $\{\ulcorner G\urcorner \mid G$ е безконтекстна граматика и $\mathcal{L}(G) = \emptyset\}$;

в) $\{\ulcorner G\urcorner \mid G$ е безконтекстна граматика над $\{0,1\}^\star$ и $\mathcal{L}(1^\star) \subseteq \mathcal{L}(G)\}$;

г) $\{\ulcorner G\urcorner \mid G$ е безконтекстна граматика над $\{0,1\}^\star$ и $\mathcal{L}(1^\star) \subseteq \mathcal{L}(G)\}$;

д) $\{\ulcorner G\urcorner \mid G$ е безконтекстна граматика над $\{0,1\}^\star$ и $\varepsilon \in \mathcal{L}(G)\}$;

е) $\{\ulcorner G\urcorner \cdot 0^k \mid G$ е безконтекстна граматика над $\{0,1\}^\star$ и $|\mathcal{L}(G)| \leq k\}$;

ж) $\{\ulcorner G\urcorner \mid G$ е безконтекстна граматика над $\{0,1\}^\star$ и $|\mathcal{L}(G)| = \infty\}$;

Задача 4.14. Докажете, че езикът

$$L = \{\ulcorner\mathcal{M}\urcorner\sharp\omega \mid \mathcal{M} \text{ прави движение наляво при работата си върху вход } \omega\}$$

е разрешим.

Упътване. Нужно е да симулирате работата на $\mathcal{M}$ върху ω само за $|\omega| + |Q^{\mathcal{M}}| + 1$ на брой стъпки. □

Задача 4.15. Докажете, че езикът съставен от думи от вида $\ulcorner\mathcal{M}\urcorner\sharp\omega$, за които $\mathcal{M}$ прави опит за движение наляво от най-лявата клетка при работата си върху вход ω е разрешим.

Библиография

[ALSU07] Alfred Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman, *Compilers: principles, techniques and tools*, 2 ed., Pearson Education, 2007.

[Brz64] Janusz A. Brzozowski, *Derivatives of regular expressions*, J. ACM **11** (1964), no. 4, 481–494.

[HMU01] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, second ed., Addison-Wesley, 2001.

[HU79] John E. Hopcroft and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, first ed., Addison-Wesley, 1979.

[Kle56] S. C. Kleene, *Representation of events in nerve nets and finite automata*, Automata Studies (C. E. Shannon and J. McCarthy, eds.), Princeton University Press, 1956, pp. 3 – 42.

[KN01] Bakhadyr Khoussainov and Anil Nerode, *Automata Theory and its Applications*, Springer, 2001.

[Koz97] Dexter Kozen, *Automata and computability*, Springer, 1997.

[Myh57] J. Myhill, *Finite automata and the representation of events*, WADD TR-57-624, Wright Patterson AFB, Ohio (1957), 112 – 137.

[Ner58] A. Nerode, *Linear automata transformation*, Proceedings of AMS **9** (1958), 541 – 544.

[NS93] Anil Nerode and Richard Shore, *Logic for Applications*, Springer-Verlag, 1993.

[PL98] Christos Papadimitriou and Harry Lewis, *Elements of the Theory of Computation*, Prentice-Hall, 1998.

[Pos46] Emil L. Post, *A variant of a recursively unsolvable problem*, Bull. Amer. Math. Soc. **52** (1946), 264–268.

[Ric53] H. G. Rice, *Classes of recursively enumerable sets and their decision problems*, Transactions of the American Mathematical Society **74** (1953), no. 2, 358–366.

[RS59] M. O. Rabin and D. Scott, *Finite automata and their decision problems*, IBM Journal of Research and Development **3** (1959), pp. 114 – 125.

[Sak09] Jacques Sakarovitch, *Elements of Automata Theory*, Cambridge University Press, 2009.

[Sha08] Jeffrey Shallit, *A Second Course in Formal Languages and Automata Theory*, CUP, 2008.

[Sip12] Michael Sipser, *Introduction to the Theory of Computation*, 3 ed., Cengage Learning, 2012.

[WWA80] L. Wittgenstein, G.H.V. Wright, and G.E.M. Anscombe, *Remarks on the philosophy of psychology*, vol. 1, University of Chicago Press/B. Blackwell, 1980.

Азбучен указател