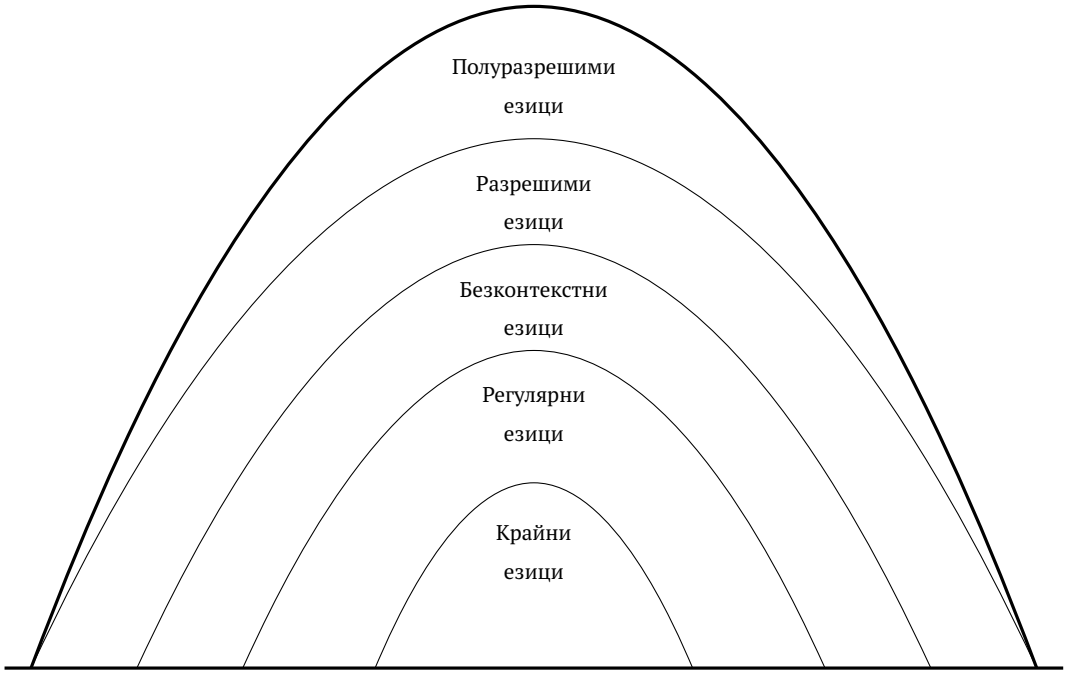
Записки по „Езици, автомати, изчислимост”

Стефан Вътев[1]

1 юни 2023 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	$\neg p$	$p \vee q$	$p \wedge q$	$p \rightarrow q$	$\neg p \vee q$	$p \Leftrightarrow q$	$(p \wedge q) \vee (\neg p \wedge \neg q)$
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 $p \rightarrow q \equiv \neg p \vee q$ и $p \Leftrightarrow q \equiv (\neg p \wedge q) \vee (p \wedge \neg q)$.

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

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

f е прекъсната в x_0 точно тогава, когато f не е непрекъсната в x_0

Да видим какво означава f да бъде *прекъсната* в точката $x_0 \in D$:

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

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.

- **Сечение**

На англ. *intersection*

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

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.

- **Обединение**

На англ. *union*

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

$A \cup B$ 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**

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

На англ. *power set*

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

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

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

Примери:

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

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

Закони на Де Морган

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

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

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

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

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

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:

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

Norbert Wiener (1914)

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

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

Kazimierz Kuratowski (1921)

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

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

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

Пример за индуктивна (рекурсивна) дефиниция

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

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

Декартово произведение

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

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

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

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:

// или f е **обратима**

- **инекция**, ако е изпълнено свойството:

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

или еквивалентно,

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

// или f е **върху** B

- **сюрекция**, ако е изпълнено свойството:

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

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

Канторово кодиране. Най-добре се вижда като се нарисува таблица

Задача 1.5. Докажете, че $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ е биекция, където

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

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

Това е малко объркващо

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:

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

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

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

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

Лесно се вижда, че $R^1 = R$

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

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

✍ Проверете, че R^+ е транзитивна релация!

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

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

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

1.4.1 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. Докажете, че за всяко $a \in \mathbb{Z}$, ако a^2 е четно, то a е четно.

Доказателство. Ние искаме да докажем твърдението P, където:

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

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

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

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

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. □

1.4.2 Индукция върху естествените числа

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

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

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

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 $P(x+1)$.

Да видим едно приложение на принципа на математическа индукция.

Задача 1.8. Докажете, че за всяко естествено число n,

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

Доказателство. Да разгледаме свойството

$$P(n) \overset{\text{деф}}{=} \sum_{i=0}^{n} 2^i = 2^{n+1} - 1.$$

Ще докажем с индукция по n, че $(\forall n)P(n)$, т.е. ще докажем следния извод:

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

- Нека първо $n = 0$. Очевидно е, че $P(0)$ е изпълнено, защото

  Това е базата на индукцията.

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

- Да разгледаме сега произволно естествено число n, като приемем, че свойството $P(n)$ е изпълнено. Ще докажем, че $P(n+1)$ също е изпълнено. Но това е лесно защото имаме следната верига от равенства:

  $P(n)$ се нарича индукционно предположение, а $P(n+1)$ се нарича индукционна стъпка.

$$\begin{aligned}
\sum_{i=0}^{n+1} 2^i &= \sum_{i=0}^{n} 2^i + 2^{n+1} \\
&= 2^{n+1} - 1 + 2^{n+1} && \text{// защото } P(n) \text{ е изпълнено} \\
&= 2.2^{n+1} - 1 \\
&= 2^{1+(n+1)} - 1 \\
&= 2^{n+2} - 1.
\end{aligned}$$

□

1.4.3 Пълна индукция върху $\mathbb{N}$

Доказателство с пълна индукция по $\mathbb{N}$ за свойството P представлява следната схема:

$$\frac{(\forall x \in \mathbb{N})[(\forall y \in \mathbb{N})[y < x \ \rightarrow P(y)] \rightarrow P(x)]}{(\forall x \in \mathbb{N})P(x)}$$

Нека да проверим принципа за пълна индукция. Да допуснем, че принципът не е верен, т.е. за някое свойство P е изпълнено, че

$$(\forall x \in \mathbb{N})[(\forall y \in \mathbb{N})[y < x \ \rightarrow P(y)] \rightarrow P(x)] \ \wedge \ (\exists x \in \mathbb{N})\neg P(x).$$

Да вземем най-малкия елемент n_0, за който $\neg P(n_0)$. От нашето допускане знаем, че такова n_0 съществува. Тогава

$$(\forall y \in \mathbb{N})[y < n_0 \ \rightarrow P(y)]$$

и следователно:

$$\frac{(\forall y \in \mathbb{N})[y < n_0 \ \rightarrow P(y)] \qquad \dfrac{(\forall x \in \mathbb{N})[(\forall y \in \mathbb{N})[y < x \ \rightarrow P(y)] \rightarrow P(x)]}{(\forall y \in \mathbb{N})[y < n_0 \ \rightarrow P(y)] \rightarrow P(n_0)} \ {\scriptstyle (x = n_0)}}{P(n_0)}$$

Така достигаме до противоречие, защото получаваме, че $P(n_0) \wedge \neg P(n_0)$.

Сега да видим, че често е полезно да използваме пълната индукция.

Задача 1.9. Докажете, че 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 $p_1, p_2, \ldots, p_k$ и естествени числа $m_1, m_2, \ldots, m_k$.

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.

□

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**.

Обикновено ще използваме малки латински букви като 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 малки гръцки букви като $\alpha, \beta, \gamma, \omega$.

- 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 $\alpha \cdot \beta \stackrel{\text{деф}}{=} (\alpha \cdot \gamma)b$.

- За две думи α и β, ще казваме, че:
  - α е **префикс** на β, ако $(\exists \gamma \in \Sigma^\star)[\, \alpha \cdot \gamma = \beta \,]$. Обикновено ще означаваме $\alpha \preceq \beta$, когато α е префикс на β.
  - α е **суфикс** на β, ако $(\exists \gamma \in \Sigma^\star)[\, \gamma \cdot \alpha = \beta \,]$.

- 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 $A^0 = \{\varepsilon\}$, $A^1 = A$, $A^2 = \{abab, abba, baba, baab\}$.
  - Съобразете, че $A^n = \{\alpha \in A^\star \mid |\alpha| = n\}$.

- 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 $L = \{0^i \mid i \in \mathbb{N}\} = \{\varepsilon, 0, 00, 000, \dots\}$. 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

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

Например, за $\alpha = abc$, то $\alpha[1] = b = \alpha[1:2]$ и $\alpha[1:3] = bc = \alpha[1:4]$, $\alpha[1:1] = \varepsilon$.

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.
$$\texttt{Pref}(L) = \{\alpha \in \Sigma^\star \mid (\exists \gamma \in \Sigma^\star)[\alpha \cdot \gamma \in L]\}.$$

- 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.
$$\texttt{Suff}(L) = \{\alpha \in \Sigma^\star \mid (\exists \gamma \in \Sigma^\star)[\gamma \cdot \alpha \in L]\}.$$

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

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

1.6 Уравнения от езици

В този раздел ще видим, че операцията звезда на Клини се оказва доста важна в нашите разглеждания.

Лема 1.1 (Ардън [2]). Нека L и M са произволни езици. Тогава езикът $L^\star \cdot M$ е *най-малкото решение* на уравнението

$$X = L \cdot X \cup M. \tag{1.1}$$

Ако $\varepsilon \notin L$, то това решение е и *единствено*.

В повечето учебници този резултат отсъства. Ние го смятаме за основен. Все пак може да се намери в някои учебници [23, стр. 100], [19, стр. 53], [14, стр. 60].

Доказателство. Първо, да видим защо $L^\star M$ е решение на уравнението (1.1). Това е лесно. Просто заместваме променливата X с езика $L^\star M$ и получаваме равенствата:

$$\begin{aligned} L^\star M &= L \cdot (L^\star M) \cup M \\ &= L^+ \cdot M \cup \{\varepsilon\} \cdot M \\ &= (L^+ \cup \{\varepsilon\}) \cdot M \\ &= L^\star \cdot M. \end{aligned}$$

Второ, да видим защо $L^\star \cdot M$ е най-малкото решение на уравнението (1.1). За целта, нека приемем, че K е произволно решение, т.е.

$$K = L \cdot K \cup M.$$

Трябва да проверим, че $L^\star \cdot M \subseteq K$. Тук ще използваме представянето

$$L^\star \cdot M = \bigcup_n L^n \cdot M.$$

Ще докажем с индукция по n, че за всяко n е изпълнено включването:

$$L^n \cdot M \subseteq K.$$

- Нека $n = 0$. Понеже $K = L \cdot K \cup M$ е ясно, че $M \subseteq K$ или с други думи, $L^0 \cdot M \subseteq K$.
- Нека сега $n > 0$. От **(И.П.)** имаме, че $L^{n-1} \cdot M \subseteq K$. Тогава, като конкатенираме с L от двете страни, получаваме:

$$\begin{aligned} L \cdot L^{n-1} \cdot M &\subseteq L \cdot K \\ &\subseteq L \cdot K \cup M \\ &= K. \end{aligned}$$

Така заключаваме, че за всяко n, е изпълнено включването $L^n \cdot M \subseteq K$ и следователно $L^\star \cdot M \subseteq K$.

Да отбележим, че ако $\varepsilon \in L$, то $\Sigma^\star$ е решение на уравнението (1.1).

Остана да докажем, че ако $\varepsilon \notin L$, то $L^\star \cdot M$ е *единственото* решение на уравнението (1.1). Ще направим това като докажем, че всяко решение K на уравнението (1.1) е такова, че $K = L^\star \cdot M$. Достатъчно да докажем, че за всяко решение K на (1.1) е изпълнено, че $K \subseteq L^\star \cdot M$, защото обратното включване $L^\star \cdot M \subseteq K$ следва от факта, че $L^\star \cdot M$ е най-малкото решение на (1.1).

Щом трябва да докажем, че $K \subseteq L^\star \cdot M$, то това означава да докажем импликацията

$$(\forall \alpha \in \Sigma^\star)[\, \alpha \in K \implies \alpha \in L^\star \cdot M \,].$$

И така, ще докажем с индукция по n, че

$$(\forall n)(\forall \alpha \in \Sigma^{\leq n})[\, \alpha \in K \implies \alpha \in L^\star \cdot M \,].$$

- Нека $n = 0$ и да вземем дума $\alpha \in \Sigma^{\leq 0}$, което означава, че $\alpha = \varepsilon$. Да приемем, че $\varepsilon \in K$, защото в противен случай импликацията би била автоматично изпълнена за $n = 0$. Щом K е решение, то $\varepsilon \in L \cdot K \cup M$, но понеже $\varepsilon \notin L$, то със сигурност $\varepsilon \in M$. Така получаваме, че

  $$\varepsilon \in K \implies \varepsilon \in L^\star \cdot M,$$

  откъдето веднага следва, че

  $$(\forall \alpha \in \Sigma^{\leq 0})[\, \alpha \in K \implies \alpha \in L^\star \cdot M \,].$$

- Нека сега $n > 0$, като приемем, че следното имаме индукционно предположение:

  $$(\forall \alpha \in \Sigma^{\leq n-1})[\, \alpha \in K \implies \alpha \in L^\star \cdot M \,].$$

  Ще докажем, че

  $$(\forall \alpha \in \Sigma^{\leq n})[\, \alpha \in K \implies \alpha \in L^\star \cdot M \,].$$

  За целта, да вземем произволна дума $\alpha \in \Sigma^{\leq n}$. Ако $|\alpha| < n$, то всъщност $\alpha \in \Sigma^{\leq n-1}$ и от **(И.П.)** веднага следва, че импликацията е изпълнена за α.

  Нека $|\alpha| = n$. Ако $\alpha \notin K$, то отново импликацията е изпълнена по тривиални причини. Интересният случай е когато $|\alpha| = n$ и $\alpha \in K$. Понеже K е решение на (1.1), то $\alpha \in L \cdot K \cup M$. Сега имаме два случая.

  – Ако $\alpha \in M$, то всичко е ясно, защото тогава $\alpha \in L^\star \cdot M$.
  – Ако $\alpha \in L \cdot K$, то $\alpha = \alpha_1 \cdot \alpha_2$, за някои думи α_1 и α_2, за които $\alpha_1 \in L$ и $\alpha_2 \in K$. Понеже $\varepsilon \notin L$, то $|\alpha_1| \geq 1$ и следователно $|\alpha_2| < |\alpha| = n$. Това означава, че от **(И.П.)** получаваме, че $\alpha_2 \in L^\star \cdot M$, откъдето заключаваме, че следното:

    $$\alpha = \alpha_1 \cdot \alpha_2 \in L \cdot (L^\star \cdot M) \subseteq L^\star \cdot M.$$

Така доказахме включването $K \subseteq L^\star \cdot M$. □

Задача 1.12. Нека L и M са произволни езици. Тогава езикът $M \cdot L^\star$ е *най-малкото* решение на уравнението

$$X = X \cdot L \cup M.$$

Ако $\varepsilon \notin L$, то това решение е и *единствено*.

[19, стр. 54]. **Задача 1.13.** Нека L_1, L_2 и M са прозволни езици. Тогава езикът $L_1^\star \cdot M \cdot L_2^\star$ е *най-малкото* решение на уравнението

$$X = L_1 \cdot X \cup X \cdot L_2 \cup M.$$

Ако $\varepsilon \notin L_1$ и $\varepsilon \notin L_2$, то това решение е и *единствено*.

1.7 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 [17], 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 фиксираме $b \in \mathbb{N}$. Да разгледаме азбуката $\Sigma = \{0, 1, \ldots, b-1\}$. Непразното множество $T \subseteq \Sigma^\star$ се нарича **дърво**, ако T изпълнява свойствата:

(1) $\texttt{Pref}(T) = T$, или с други думи,

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

Това свойство казва, че T е затворено относно префикси.

(2) За всяка дума $\alpha \in \Sigma^\star$ и $x \in \Sigma$,

$$\alpha \cdot x \in T \,\&\, y < x \implies \alpha \cdot y \in T.$$

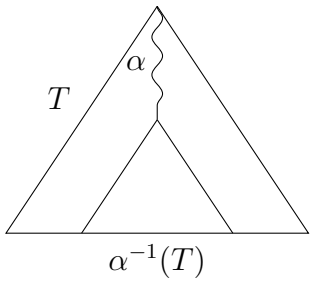
Едно от удобствата да се работи с тази дефиниция на дърво е, че ако имаме един връх от дървото, т.е. дума $\alpha \in T$, то тя пази информация за целия път от корена до този връх. Например, ако $010 \in T$, то от корена първо сме отишли по левия клон, после по десния, и най-накрая пак по левия. Фактът, че според тази дефиниция имаме подредба на върховете на дървото ще ни бъде полезен по-нататък. Например, 1011 е по-наляво от 110, защото лексикографски 1011 по-малка от 110.

Ето няколко примера:

- $T = \{\varepsilon, 0, 1, 00, 01, 11\}$ не е дърво, защото (2) не е удовлетворено.
- $T = \{\varepsilon, 0, 1, 00, 01, 10\}$ е дърво.
- $T = \{0\}^\star$ е безкрайно дърво, докато $T = \{1\}^\star$ не е.
- $T = \{0\}^\star \cdot \{\varepsilon, 1\}$ е безкрайно дърво.
- $T = \{0, 1\}^\star$ е пълното двоично дърво.

Нека да въведем следните означения:

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



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

Задача 1.14. Докажете, че ако T е дърво, то $\texttt{Pref}(\texttt{leaves}(T)) = T$.

Задача 1.15. 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.

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

□

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

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 (ДКА) е петорка

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

където

- 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 $\alpha \in \Sigma^\star$, където $\alpha = a_0 a_1 \cdots a_{n-1}$. 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. $L = \{\alpha \in \Sigma^\star \mid \mathcal{A} \text{ разпознава } \alpha\}$. В такъв случай ще казваме, че езикът L е **автоматен**.

Algorithm 1 Проблемът за принадлежност за автоматни езици

```
1: procedure BELONG(A, ω)
2:     q := q_start
3:     for all i < |ω| do
4:         q := δ(q, ω[i])
5:     if q ∈ F then
6:         return True
7:     else
8:         return False
```

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:

Възможно е да се дефинира $\delta^\star$ и така:

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

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

✍ Съобразете го сами!

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

Определение 2.2. За произволен ДКА $\mathcal{A}$, дефинираме

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

Понякога е удобно да разгледаме произволно състояние на автомата $\mathcal{A}$ като начално и да разгледаме съответния език, който получаваме. Поради тази причина полагаме за произволно $q \in Q$,

$$\mathcal{L}_{\mathcal{A}}(q) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \delta^\star(q, \omega) \in F\}.$$

В частност, $\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}})$.

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

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

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

Упътване. Индукция по дължината на думата β.

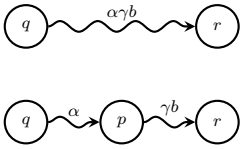
- $|\beta| = 0$, т.е. $\beta = \varepsilon$. Тогава за всяко състояние q и произволна дума α имаме:

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

- 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:

Индукционната стъпка може да се представи и така:



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

□

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:

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

Сега ще видим една естествена характеризация на езиците $\mathcal{L}_\mathcal{A}(q)$ като решение на система от уравнения.

[23, стр. 133], [14, стр. 63].

Твърдение 2.2 (Фундаментално свойство на автоматните езици)**.** Да разгледаме произволен ДКА $\mathcal{A}$ и нека за улесние да индексираме състоянията така:

$$Q = \{q_1, q_2, \ldots, q_n\}.$$

За произволни индекси $i, j = 1, 2, \ldots, n$, да положим:

$$R_{i,j} \stackrel{\text{деф}}{=} \{a \in \Sigma \mid \delta(q_i, a) = q_j\}$$
$$L_i \stackrel{\text{деф}}{=} \mathcal{L}_{\mathcal{A}}(q_i)$$
$$P_i \stackrel{\text{деф}}{=} \begin{cases} \{\varepsilon\}, & \text{ако } q_i \in F \\ \emptyset, & \text{ако } q_i \notin F. \end{cases}$$

Тогава имаме следното:

$$\left|\begin{array}{rcl} L_1 & = & R_{1,1} \cdot L_1 \cup \cdots R_{1,n} \cdot L_n \cup P_1 \\ & \vdots & \\ L_n & = & R_{n,1} \cdot L_1 \cup \cdots R_{n,n} \cdot L_n \cup P_n. \end{array}\right.$$

С други думи, езиците $L_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ са решение на системата:

$$\left|\begin{array}{rcl} X_1 & = & R_{1,1} \cdot X_1 \cup \cdots R_{1,n} \cdot X_n \cup P_1 \\ & \vdots & \\ X_n & = & R_{n,1} \cdot X_1 \cup \cdots R_{n,n} \cdot X_n \cup P_n. \end{array}\right.$$

Обърнете внимание, че все още не знаем дали $L_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ са *единственото* решение на системата. Това ще видим чак в *Теорема 2.4*.

Упътване. Можем да разгледаме всеки ред поотделно. Достатъчно е да докажем, че за произволно i, където $1 \leq i \leq n$, е изпълнено равенството

$$\mathcal{L}_{\mathcal{A}}(q_i) = \bigcup_{j=1}^{n} R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j) \cup P_i.$$

Първо, за включването ($\subseteq$), нека разгледаме дума ω, за която

$$\omega \in \mathcal{L}_{\mathcal{A}}(q_i).$$

- Нека $\omega = \varepsilon$. Ясно е, че $q_i \in F$. Тогава, по дефиниция, $\omega \in P_i$.
- Нека $|\omega| > 0$. Тогава думата ω може да се запише във вида $\omega = b\alpha$. Щом $\omega \in \mathcal{L}_{\mathcal{A}}(q_i)$, то $\delta^\star_{\mathcal{A}}(q_i, b\alpha) \in F_{\mathcal{A}}$. Това означава, че $\delta^\star_{\mathcal{A}}(\delta_{\mathcal{A}}(q_i, b), \alpha) \in F_{\mathcal{A}}$. Нека $\delta_{\mathcal{A}}(q_i, b) = q_j$. Тогава $\alpha \in \mathcal{L}_{\mathcal{A}}(q_j)$ и $b \in R_{i,j}$. Заключаваме, че

$$\underbrace{b\alpha}_{\omega} \in R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j).$$

Второ, за включването ($\supseteq$), нека разгледаме дума ω, за която

$$\omega \in \bigcup_{j=1}^{n} R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j) \cup P_i.$$

- Нека $\omega \in P_i$. Тогава $\omega = \varepsilon$. Ясно е, по дефиниция на P_i, че $q_i \in F$. Ясно е, че $\omega \in \mathcal{L}_{\mathcal{A}}(q_i)$.
- Нека $\omega \in R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j)$, за някое j, където $1 \leq j \leq n$. Тогава ω може да се представи като $\omega = b\alpha$, където $b \in R_{i,j}$ и $\alpha \in \mathcal{L}_{\mathcal{A}}(q_j)$. Заключаваме, че $\delta_{\mathcal{A}}(q_i, b) = q_j$ и $\delta^{\star}_{\mathcal{A}}(q_j, \alpha) \in F_{\mathcal{A}}$. Накрая, $\delta^{\star}_{\mathcal{A}}(q_i, \omega) \in F_{\mathcal{A}}$ и тогава можем да заключим, че
$$\omega \in \mathcal{L}_{\mathcal{A}}(q_i).$$

□

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:

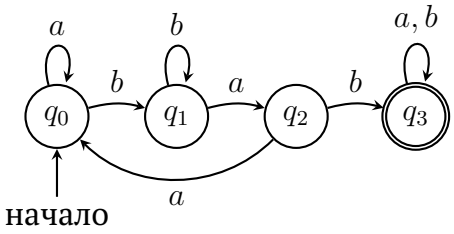
$$\mathcal{L}(\mathcal{A}) = \{\alpha \in \Sigma^{\star} \mid (\exists f \in F)[(q_{\mathtt{start}}, \alpha) \vdash^{\star}_{\mathcal{A}} (f, \varepsilon)]\}.$$

Примерни задачи

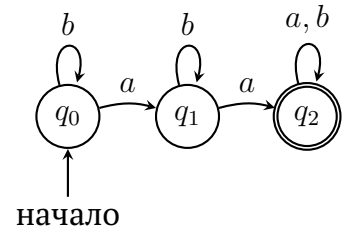
В този раздел ще разгледаме няколко примера за автоматни езици и ще видим как можем да докажем, че конкретен автомат разпознава даден език.

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

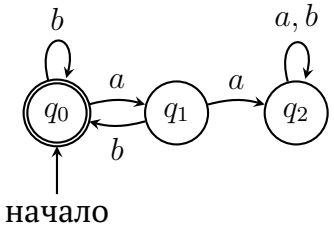
Пример 2.1. Да разгледаме няколко примера за автомати и езиците, които разпознават. Дефинирайте функцията на преходите δ за всеки автомат.



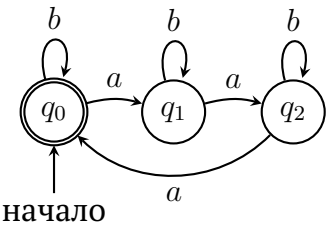
(а) $\{\omega \in \{a,b\}^\star \mid \omega \text{ съдържа } bab\}$



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

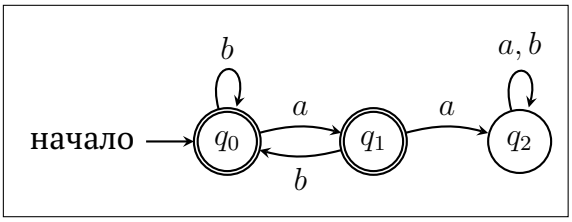
В повечето от горните примери може сравнително лесно да се съобрази, че построеният автомат разпознава желания език. При по-сложни задачи, обаче, ще се наложи да дадем доказателство, като обикновено се прилага *метода на математическата индукция* върху дължината на думите. Ще разгледаме няколко такива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

Задача 2.2. Докажете, че езикът L е автоматен, където:

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

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 **??**. По-късно ще разгледаме общ метод за строене на автомат по език.

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

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

Да напомним, че $|\alpha| \stackrel{\text{деф}}{=}$ дължината на α.

(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 включването

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

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

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

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

което, разгледана в контрапозиция, е еквивалентна на

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

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

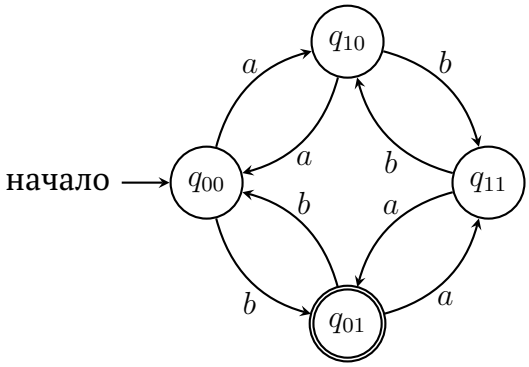
Да напомним, че $|\omega|_a \stackrel{\text{деф}}{=}$ броят на срещанията на буквата a в думата ω.

Задача 2.3. Докажете, че езикът L е автоматен, където:

$$L \stackrel{\text{деф}}{=} \{\omega \in \{a, b\}^\star \mid |\omega|_a \equiv 0 \bmod 2 \;\&\; |\omega|_b \equiv 1 \bmod 2\}.$$

Тази задача е важна, защото в явен вид показва как кодираме информация в състоянията на автомата, а именно остатъците при деление на 2 на $|\omega|_a$ и $|\omega|_b$.

Упътване. Разгледайте детерминирания автомат $\mathcal{A}$:



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

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,

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

За $k = 2$, можем да изразим $\overline{\alpha}_{(2)}$ рекурсивно така:

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

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

$$\begin{aligned}\overline{10}_{(2)} &= \overline{010}_{(2)} \\ &= \overline{0010}_{(2)} \\ &= \cdots\end{aligned}$$

Задача 2.4. Докажете, че езикът L е автоматен, където:

$$L \stackrel{\text{деф}}{=} \{\alpha \in \{0,1\}^\star \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\}.$$

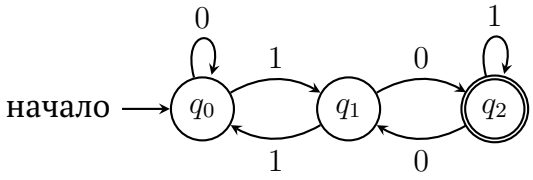
Доказателство. Нашият автомат ще има три състояния $\{q_0, q_1, q_2\}$, като началното състояние ще бъде q_0. Целта ни е да дефинираме така автомата, че да имаме следното свойство:

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

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

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

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 (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. $\alpha = \beta x$, $|\beta| = n$.

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).

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

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

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

Затвореност относно булеви операции

В този раздел ще видим, че автоматните езици са затворени относно основните булеви операции над езици - обединение, сечение и разлика.

Твърдение 2.3. 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:

- $Q \stackrel{\text{деф}}{=} Q_1 \times Q_2$;
- За всяко $\langle r_1, r_2 \rangle \in Q$ и всяко $a \in \Sigma$,

Съобразете, че δ е тотална функция.

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

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

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:

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

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:

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

□

Твърдение 2.4. 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.

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

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

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 $\mathcal{A}$. □

✍ 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.

Твърдение 2.5. 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.

Упътване. Първият подход е да използваме конструкцията на автомата $\mathcal{A}$ от *Твърдение 2.3*, с единствената разлика, че тук избираме финалните състояния да бъдат елементите на множеството

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

✍ 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.

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

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

□

Като приложение на наученото тук, нека отново да разгледаме *Задача 2.3*.

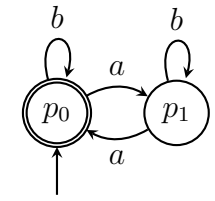
Задача 2.5. Докажете, че езикът L е автоматен, където:

$$L = \{\omega \in \{a,b\}^\star \mid |\omega|_a \equiv 0 \bmod 2 \ \& \ |\omega|_b \equiv 1 \bmod 2\}.$$

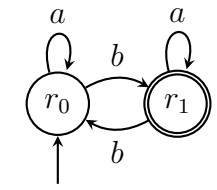
Упътване. Нека да представим езика L като $L = L_1 \cap L_2$, където

$$L_1 = \{\omega \in \{a,b\}^\star \mid |\omega|_a \equiv 0 \bmod 2\}$$
$$L_2 = \{\omega \in \{a,b\}^\star \mid |\omega|_b \equiv 1 \bmod 2\}.$$

Да разгледаме следните детерминирани крайни автомати $\mathcal{A}_1$ и A_2:



(а) $\mathcal{L}(\mathcal{A}_1) = L_1$

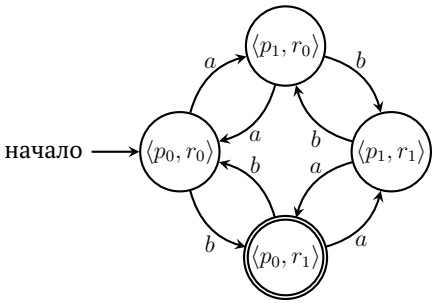


(б) $\mathcal{L}(\mathcal{A}_2) = L_2$

✍ Съобразете сами, че $\mathcal{L}(\mathcal{A}_1) = L_1$ и $\mathcal{L}(\mathcal{A}_2) = L_2$!

Ясно е, че $L = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$. Ще построим ДКА $\mathcal{A}$ като използваме конструкцията от *Твърдение 2.3*. Така финалното състояние на $\mathcal{A}$ ще бъде $\langle p_0, r_1\rangle$, защото p_0 и r_1 са единствените финални състояния съответно на $\mathcal{A}_1$ и $\mathcal{A}_2$. Началното състояние на $\mathcal{A}$ ще бъде $\langle p_0, r_0\rangle$, защото p_0 е началното състояние на A_1 и r_0 е началното състояние на $\mathcal{A}_2$. Сега, използвайки, че $\delta(\langle p_i, r_j\rangle, x) = \langle \delta_1(p_i, x), \delta_2(r_j, x)\rangle$, получаваме следния автомат $\mathcal{A}$.

Ако означим състоянията $\langle p_i, r_j\rangle$ като q_{ij}, то ще получим точно автомата от *Задача 2.3*.



Фигура 2.8: $\mathcal{L}(\mathcal{A}) = \{\omega \in \{a,b\}^\star \mid |\omega|_a \equiv 0 \bmod 2 \ \& \ |\omega|_b \equiv 1 \bmod 2\}$

□

2.2 Недетерминирани крайни автомати

В този раздел ще разгледаме едно обобщение на детерминираните крайни автомати, при които ще позволяваме от дадено състояние и дадена буква да отиваме в много различни състояния, а не само едно.

Определение 2.3. Недетерминиран краен автомат представлява петорка от вида

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

където

Въведени от Рабин и Скот [21]. За по-голяма яснота, често ще означаваме с $\mathcal{N}$ недетерминирани автомати, като ще запазим $\mathcal{A}$ за детерминираните. Мотивация - [13, 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 $\Delta(q, a) = \emptyset$;
- $Q_{\texttt{start}} \subseteq Q$ е множество от начални състояния;
- $F \subseteq Q$ е множеството от финални състояния.

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

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 множество от състояния $R \subseteq Q$ и дума $\alpha \in \Sigma^\star$ по следния начин:

В [18] Δ е релация и се позволяват ε-преходи.
В [25] пък е функция, но пак се позволяват ε-преходи.
В [10] 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
$$\Delta^\star(R, \beta a) \stackrel{\text{деф}}{=} \bigcup \{\Delta(p, a) \mid p \in \Delta^\star(R, \beta)\}.$$

Дефиницията на език разпознаван от недерерминиран краен автомат $\mathcal{N}$ е малко по-сложна от тази за ДКА.

Понеже операцията $\bigcup$ е сложна, 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 $R \subseteq Q$,

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

Сравенете с *Твърдение 2.1*.

✍ Трябва да може да докажете това твърдение сами!

Упътване. Индукция по дължината на β. □

Доказателство. Ще докажем, че $(\forall n)P(n)$ с индукция по n, където

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

- 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:

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

Получаваме, че

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

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

Теорема 2.1 (Рабин-Скот [21]). За всеки НКА $\mathcal{N}$ съществува еквивалентен на него ДКА $\mathcal{D}$, 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

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

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

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

$$\delta(\underbrace{R}_{\text{състояние}}, a) \overset{\text{деф}}{=} \Delta^\star(\underbrace{R}_{\text{множество}}, a).$$

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

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

$$\Delta^\star(R,a) = \bigcup_{q \in R} \Delta(q,a).$$

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

$$\Delta^\star(\underbrace{R}_{\text{множество}}, \alpha) = \delta^\star(\underbrace{R}_{\text{състояние}}, \alpha). \tag{2.4}$$

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 $\delta^\star$, че за всяко $R \subseteq Q$ е изпълнено:
$$\Delta^\star(R,\varepsilon) = R = \delta^\star(R,\varepsilon).$$
- Да приемем, че (2.4) е изпълнено за думи α с дължина n, т.е.
$$(\forall \alpha \in \Sigma^n)(\forall R \subseteq Q)[\, \Delta^\star(R,\alpha) = \delta^\star(R,\alpha)\,].$$

- Нека сега α има дължина $n+1$, т.е. $\alpha = \beta a$, където $|\beta| = n$ и $a \in \Sigma$.

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

Така доказахме, че

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

Това означава, че според принципа на математическата индукция имаме Свойство (2.4).

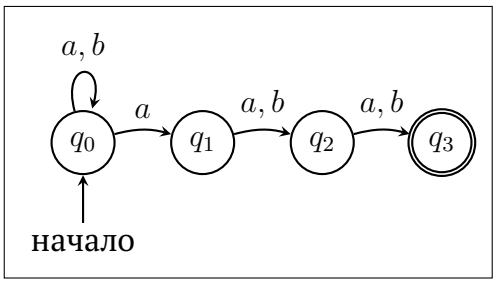
Сега вече е лесно да съобразим, че

$$\begin{aligned}
\omega \in \mathcal{L}(\mathcal{D}) &\Leftrightarrow \delta^\star(q_{\texttt{start}}, \omega) \in F' && \text{// деф. на } \mathcal{L}(\mathcal{D}) \\
&\Leftrightarrow \Delta^\star(Q_{\texttt{start}}, \omega) \cap F \neq \emptyset && \text{// от (2.4)} \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{N}) && \text{// деф. на } \mathcal{L}(\mathcal{N}).
\end{aligned}$$

□

Хубаво е да има един пример за детерминизация.

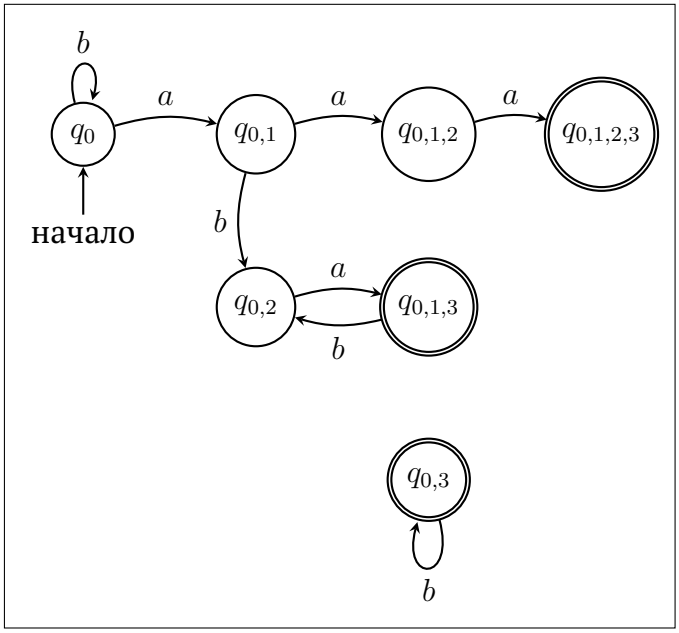
Пример 2.2. Да разгледаме следния недетерминиран краен автомат $\mathcal{N}$ зададен на *Фигура 2.12*.



Фигура 2.11: Недетерминиран автомат $\mathcal{N}$ за езика съставен думите, в които има a на трета позиция от дясно на ляво.

- $q_{\texttt{start}} = \{q_0\}$

-



Фигура 2.12: Недетерминиран автомат $\mathcal{N}$ за езика съставен думите, в които има a на трета позиция от дясно на ляво.

Задача 2.6. Докажете, че автоматните езици са затворени относно операцията `rev`. С други думи, докажете, че ако L е автоматен език, то $L^{\mathtt{rev}} \stackrel{\text{деф}}{=} \{\omega^{\mathtt{rev}} \mid \omega \in L\}$ 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 $\mathcal{L}(\mathcal{N}) = L^{\mathtt{rev}}$.

- $Q^{\mathcal{N}} \stackrel{\text{деф}}{=} Q^{\mathcal{A}}$;
- $Q^{\mathcal{N}}_{\mathtt{start}} \stackrel{\text{деф}}{=} F^{\mathcal{A}}$;
- $\Delta_{\mathcal{N}}(q, a) \stackrel{\text{деф}}{=} \{p \in Q^{\mathcal{A}} \mid \delta_{\mathcal{A}}(p, a) = q\}$;
- $F^{\mathcal{N}} \stackrel{\text{деф}}{=} \{q_{\mathtt{start}}\}$.

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

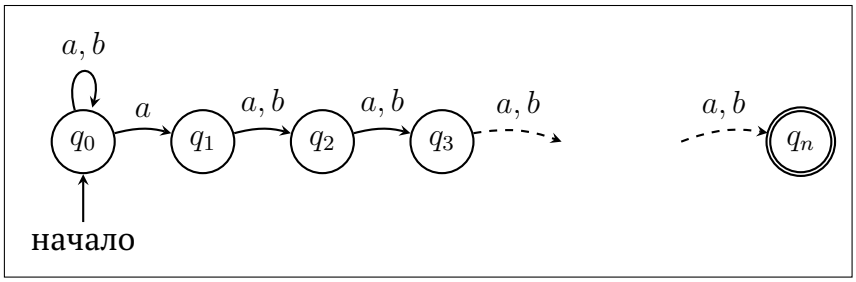
2.2.1 Експоненциална експлозия

Тук следваме [11, стр. 66] и [1, стр. 164]. В [24, стр. 80] има друг пример за НКА с n състояния вместо $n+1$, но доказателството изглежда по-сложно.

Сега ще видим, че всеки алгоритъм за детерминизация е експоненциален по време и памет.

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 $\mathcal{N}$ зададен на *Фигура 2.13*.



Фигура 2.13: Недетерминиран автомат $\mathcal{N}$ за езика съставен думите, в които има a на n-та позиция от дясно на ляво.

Лесно се съобразява, че за $n > 0$, недетерминирният краен автомат $\mathcal{N}$ на *Фигура 2.13* с $n+1$ на брой състояния разпознава езика

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

Езикът L може да се представи и така:

$$L = \{a,b\}^\star \cdot \{a\} \cdot \{a,b\}^{n-1}.$$

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:

$$\alpha = \lambda \cdot \alpha[i] \cdot \rho \text{ и } \beta = \lambda \cdot \beta[i] \cdot \rho'.$$

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

Тогава $\alpha_0 = \lambda \cdot a \cdot \rho_0$, където $|\rho_0| = n - 1$. Аналогично, $\beta_0 = \lambda \cdot b \cdot \rho_1$, където $|\rho_1| = n - 1$. 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.

□

2.3 Регулярни езици

Да фиксираме една непразна азбука Σ. **Регулярните изрази** r могат да се опишат със следната граматика:

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

Това е пример за индуктивна дефиниция

Регулярните изрази могат да се опишат и по следния начин:

- Символите $\boldsymbol{\emptyset}$, $\boldsymbol{\varepsilon}$ са регулярни изрази;
- за всяка буква $a \in \Sigma$, символът $\boldsymbol{a}$ е регулярен израз;

В литературата също се среща записът $(\boldsymbol{r}_1 \mid \boldsymbol{r}_2)$ вместо $(\boldsymbol{r}_1 + \boldsymbol{r}_2)$

- ако $\boldsymbol{r_1}$ и $\boldsymbol{r_2}$ 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;
- Всеки регулярен израз е получен по някое от горните правила.

Това е друг пример за индуктивна (рекурсивна) дефиниция.

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 $\boldsymbol{r}$ ще определим език $\mathcal{L}[\![\boldsymbol{r}]\!]$.

- $\emptyset$ 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 $\boldsymbol{a}$. Означаваме

$$\mathcal{L}[\![\boldsymbol{a}]\!] \stackrel{\text{деф}}{=} \{a\};$$

Понякога, когато приоритетът на операциите е ясен, ще изпускаме да пишем скоби.

- Нека L_1 и L_2 са регулярни езици, т.е. съществуват регулярни изрази $\boldsymbol{r}_1$ и $\boldsymbol{r}_2$, за които $\mathcal{L}[\![\boldsymbol{r_1}]\!] = L_1$ и $\mathcal{L}[\![\boldsymbol{r_2}]\!] = L_2$. Тогава:
  - $L_1 \cup L_2$ е регулярен език, който се описва с регулярния израз $(\boldsymbol{r}_1 + \boldsymbol{r}_2)$. Тогава
  $$\mathcal{L}[\![(\boldsymbol{r_1} + \boldsymbol{r_2})]\!] \stackrel{\text{деф}}{=} \mathcal{L}[\![\boldsymbol{r_1}]\!] \cup \mathcal{L}[\![\boldsymbol{r_2}]\!].$$

Тази операция се нарича конкатенация. Обикновено изпускаме знака $\cdot$

  - $L_1 \cdot L_2$ е регулярен език, който се описва с регулярния израз $(\boldsymbol{r}_1 \cdot \boldsymbol{r}_2)$. Тогава
  $$\mathcal{L}[\![(\boldsymbol{r_1} \cdot \boldsymbol{r_2})]\!] \stackrel{\text{деф}}{=} \mathcal{L}[\![\boldsymbol{r_1}]\!] \cdot \mathcal{L}[\![\boldsymbol{r_2}]\!].$$

Звезда на Клини

  - $L_1^\star$ е регулярен език, който се описва с регулярния израз $\boldsymbol{r_1^\star}$. Тогава

$$\mathcal{L}[\![\boldsymbol{r_1^\star}]\!] \stackrel{\text{деф}}{=} \mathcal{L}[\![\boldsymbol{r_1}]\!]^\star.$$

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.

Пример 2.3. Нека да построим регулярни изрази за всеки от езиците от *Пример 2.1*.

В [25, 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.

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

$$\mathcal{L}[\![\boldsymbol{r}]\!] = \{\omega \in \{a,b\}^\star \mid \omega \text{ съдържа } bab\}.$$

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

$$\mathcal{L}[\![\boldsymbol{r}]\!] = \{\omega \in \{a,b\}^\star \mid |\omega|_a \geq 2\}.$$

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

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

$$\mathcal{L}[\![\boldsymbol{r}]\!] = \{\omega \in \{a,b\}^\star \mid \text{ всяко } a \text{ в } \omega \text{ се следва от поне едно } b\}.$$

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

$$\mathcal{L}[\![\boldsymbol{r}]\!] = \{\omega \in \{a,b\}^\star \mid |\omega|_a \equiv 0 \bmod 3\}.$$

Нека да въведем следните означения, които ще олеснят записа на регулярни изрази.

Определение 2.4. За произволни регулярни изрази $\boldsymbol{r}$ и $\boldsymbol{s}$, ще използваме записа $\boldsymbol{r} = \boldsymbol{s}$, ако $\mathcal{L}[\![\boldsymbol{r}]\!] = \mathcal{L}[\![\boldsymbol{s}]\!]$.

Задача 2.7. Докажете, че са изпълнени следните равенства:

а) $(\boldsymbol{rs})^\star\boldsymbol{r} = \boldsymbol{r}(\boldsymbol{sr})^\star$;

б) $(\boldsymbol{r}+\boldsymbol{s})^\star = \boldsymbol{r}^\star(\boldsymbol{sr}^\star)^\star$

Задача 2.8. Проверете дали са изпълнени следните равенства:

а) $r + s = s + r$;

б) $(\varepsilon + r)^\star = r^\star$;

в) $(r^\star s^\star)^\star = (r + s)^\star$;

г) $(r^\star)^\star = r^\star$;

д) $r^\star r^\star = r^\star$;

е) $(rs + r)^\star r = r(sr + r)^\star$;

ж) $s(rs + s)^\star r = rr^\star s(rr^\star s)^\star$;

з) $(r + s)^\star = r^\star + s^\star$;

и) $(r + s)^\star s = (r^\star s)^\star$;

к) $(rs + r)^\star rs = (rr^\star s)^\star$;

л) $\emptyset^\star = \varepsilon$;

м) $rs = st \implies r^\star s = st^\star$;

н) $r^\star = (\varepsilon + r)^n (r^n)^\star$, за всяко $n \in \mathbb{N}$.

2.4 Регулярните езици са автоматни (автоматен подход)

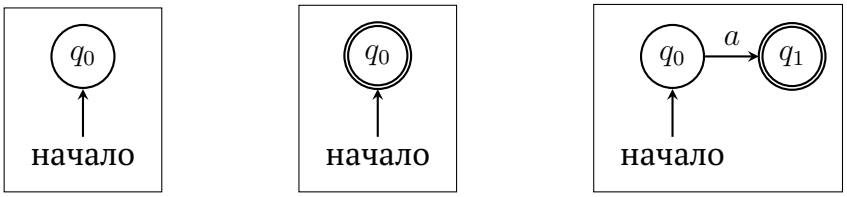
В този раздел ще покажем, че всеки регулярен език се разпознава с НКА. Нашият подход ще бъде конструктивен. Алгебричният подход към доказателството на този резултат ще изложим в Раздел 2.8. Доказателство ни ще протече с индукция по построението на регулярните езици. Да започнем с базата на индукцията.

Твърдение 2.8. 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)** $\mathcal{L}(\mathcal{N}) = \{a\}$

□

Сега нека да преминем към индуционната стъпка. Имаме три случая, които трябва да разгледаме.

Лема 2.1. 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.

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

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

- $\mathcal{A}_1 = \langle \Sigma, Q_1, \delta_1, q'_{\texttt{start}}, F_1 \rangle$, 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

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

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

Идеята в тази конструкция е, че свързваме финалните състояния на първия автомат с наследниците на началносто състояние на втория автомат.

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:

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

Сега съединяваме и изчислението за α и получаваме, че:

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

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 $|\omega| = n$. 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:

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

Нека $\omega_1 = \omega[:\ell]$ и $\omega_2 = \omega[\ell:]$. Ясно е, че:

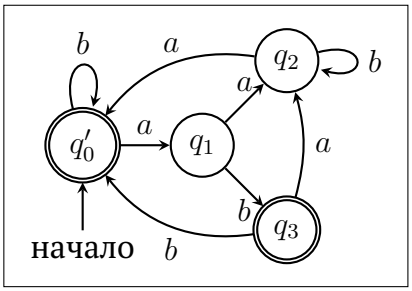
$$(q_0, \omega) \vdash^\star_{\mathcal{N}} (q_\ell, \omega[\ell:]) \vdash_{\mathcal{N}} (q_{\ell+1}, \omega[\ell+1:]) \vdash^\star_{\mathcal{N}} (q_n, \varepsilon).$$

От конструкцията на $\mathcal{N}$ следва, че редицата от състояния $(q_i)_{i=0}^\ell$ описва приемащо изчисление на $\mathcal{A}_1$ върху ω_1. 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:

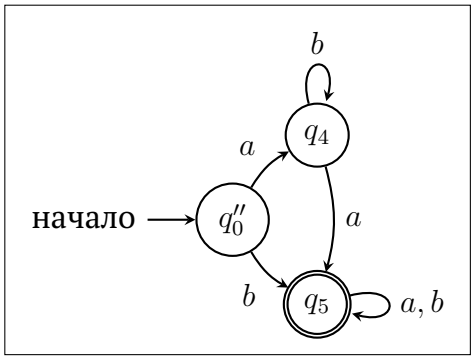
Това е единственият начин да направим преход от състояние на Q_1 към състояние на Q_2.

- $(q_0, \omega_1) \vdash^\star_{\mathcal{A}_1} (q_\ell, \varepsilon)$. Понеже $q_0 = q'_{\texttt{start}}$ и $q_\ell \in F_1$, то $\omega_1 \in \mathcal{L}(\mathcal{A}_1)$.
- $(q''_{\texttt{start}}, \omega_2) \vdash^\star_{\mathcal{A}_2} (q_n, \varepsilon)$. Понеже $q_n \in F_2$, то $\omega_2 \in \mathcal{L}(\mathcal{A}_2)$.

□

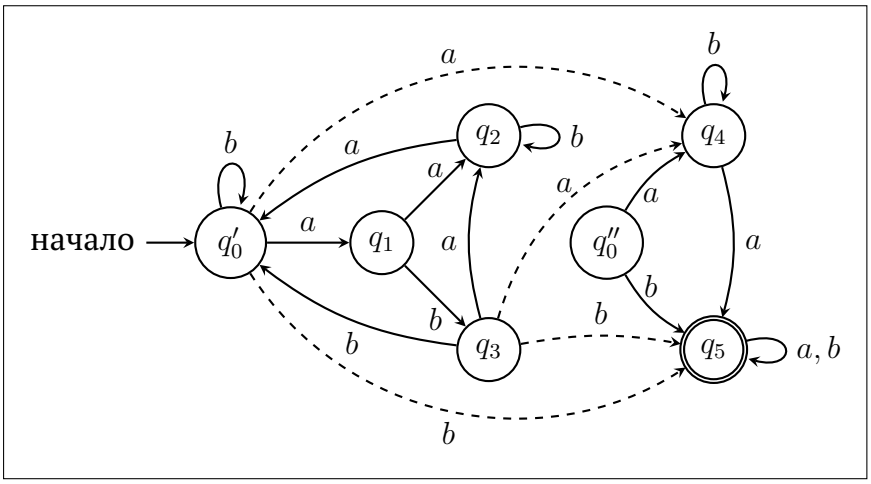


(а) автомат $\mathcal{A}_1$



(б) автомат $\mathcal{A}_2$

Пример 2.4. 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 2.16: $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A}_1) \cdot \mathcal{L}(\mathcal{A}_2)$

[23, стр. 146].

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 от q_0' състояние, но в общия случай не можем да го премахнем, защото може да има преходи влизащи в q_0''.

Лема 2.2. Класът от автоматните езици е затворен относно операцията *обединение*.

Да напомним, че вече знаем, че автоматните езици са затворени относно операцията обединие. Това видяхме в *Твърдение 2.5*. Сега ще дадем втора конструкция.

Упътване. Нека са дадени детерминистичните автомати:

- $\mathcal{A}_1 = \langle \Sigma, Q_1, \delta_1, q'_{\texttt{start}}, F_1 \rangle$, като $L(\mathcal{A}_1) = L_1$;
- $\mathcal{A}_2 = \langle \Sigma, Q_2, \delta_2, q''_{\texttt{start}}, F_2 \rangle$, като $L(\mathcal{A}_2) = L_2$.

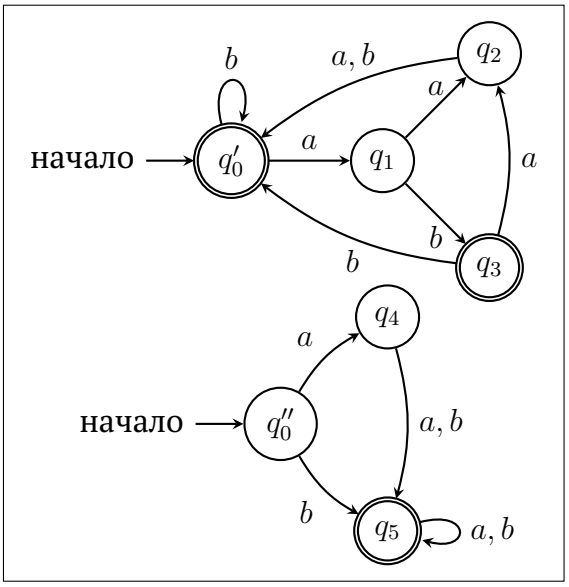
Ще дефинираме автомата $\mathcal{N} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle$, така че

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

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

□

Пример 2.5. 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 2.17: $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

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 2.3. 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

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

Първо ще построим недетерминиран краен автомат

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

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:

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

Идеята в тази конструкция е, че свързваме финалните състояния на автомата с наследниците на началното състояние.

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 $\alpha = \alpha_0 \alpha_1 \cdots \alpha_\ell$, 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 $\alpha \in \mathcal{L}^{\ell+1}(\mathcal{A})$, за някое ℓ, и следователно думата α може да се представи като $\alpha = \alpha_0 \cdot \alpha_1 \cdot \cdots \alpha_\ell$, 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:

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

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

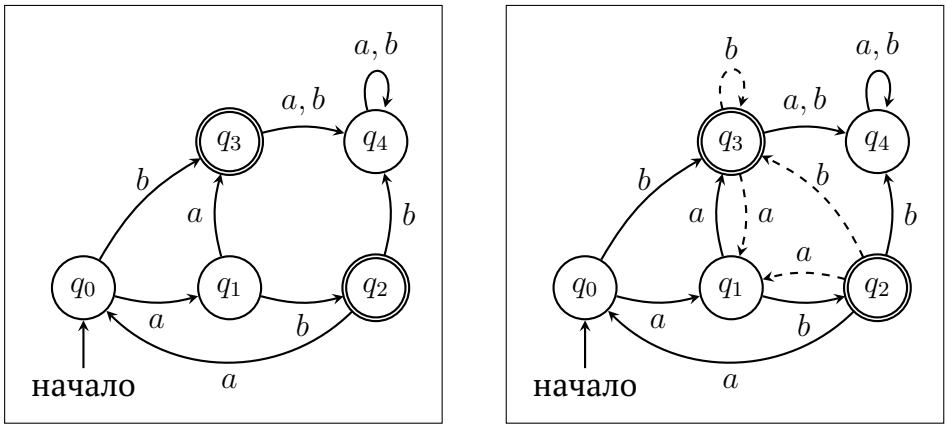
Така доказахме, че $\mathcal{L}(\mathcal{N}) = (\mathcal{L}(\mathcal{A}))^+$. Сега ще построим недетерминиран краен автомат $\mathcal{N}' = \langle \Sigma, Q', Q'_{\texttt{start}}, \Delta, F' \rangle$, такъв че:

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

- $Q' \stackrel{\text{деф}}{=} Q \cup \{q'_{\texttt{start}}\}$;
- $F' \stackrel{\text{деф}}{=} F \cup \{q'_{\texttt{start}}\}$;
- $Q'_{\texttt{start}} = Q_{\texttt{start}} \cup \{q'_{\texttt{start}}\}$;

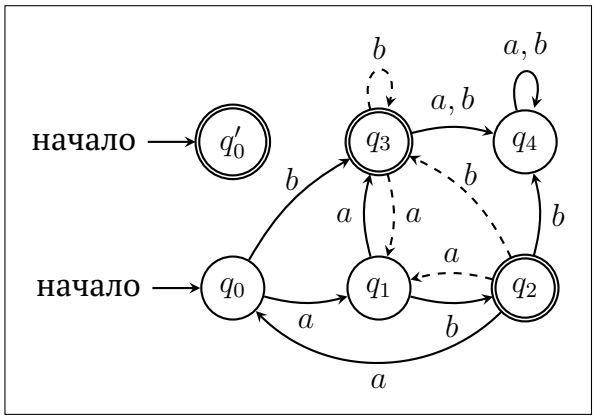
Лесно се съобразява, че $\mathcal{L}(\mathcal{N}') = \mathcal{L}(\mathcal{N}) \cup \{\varepsilon\}$. □

Пример 2.6. 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 2.19: $\mathcal{L}(\mathcal{N}') = \mathcal{L}(\mathcal{N}) \cup \{\varepsilon\} = \mathcal{L}(\mathcal{A})^\star$

Сега можем да обобщим всичко дотук в следната жизненоважна теорема.

Теорема 2.2 (Клини[12])**.** Всеки регулярен език е автоматен.

2.5 Автоматните езици са регулярни (автоматен подход)

Нашата цел е да докажем, че автоматните езици съвпадат с регулярните. За целта, ще разгледаме двете посоки на тази задача поотделно. В този раздел докажем, че всеки автоматен език е регулярен, а после ще видим, че всеки регулярен език е автоматен. Алтернативно доказателство на тази теорема е изложено в Раздел 2.6, където езикът на автомата се представя като решение на система от уравнения.

Тук трябва да си представим автомата като един граф и да разгледаме пътищата между всеки два върха в графа.

Теорема 2.3 (Клини [12]). Всеки автоматен език е регулярен.

Това е класическото доказателство на тази теорема [18, стр. 79], [10, стр. 33]. Аз предпочитам доказателството в *Раздел 2.6*.

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

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

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

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

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

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

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

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
$$L(i,j,0) = \{\varepsilon\} \cup \{a \in \Sigma \mid \delta(q_i, a) = q_j\}. \tag{2.5}$$
- 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 $\boldsymbol{r^k_{i,j}}$, които описват езиците $L(i,j,k)$, т.е. имаме индукционното предположение $P(k)$, за което

$$(\forall i < n)(\forall j < n)[L(i,j,k) = \mathcal{L}[\![\boldsymbol{r^k_{i,j}}]\!]].$$

Ще докажем $P(k+1)$, т.е. съществуват регулярни изрази $\boldsymbol{r^{k+1}_{i,j}}$, такива че

$$(\forall i < n)(\forall j < n)[\, L(i,j,k+1) = \mathcal{L}[\![\boldsymbol{r^{k+1}_{i,j}}]\!]\,].$$

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

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

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:

✍ Защо?

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

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

$$\boldsymbol{r_{i,j}^{k+1}} = \boldsymbol{r_{i,j}^{k}} + \boldsymbol{r_{i,k}^{k}} \cdot (\boldsymbol{r_{k,k}^{k}})^{\star} \cdot \boldsymbol{r_{k,j}^{k}}. \tag{2.6}$$

Така доказахме $P(k+1)$.

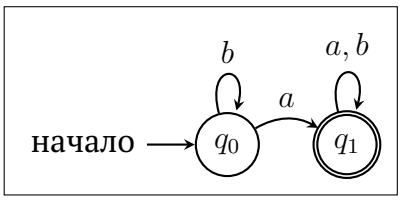
Заключаваме, че за всеки два индекса $i, j < |Q|$ и индекс $k \leq |Q|$, езикът $L(i, j, k)$ може да се опише с регулярен израз $\boldsymbol{r_{i,j}^{k}}$. Тогава ако $F = \{q_{i_1}, \ldots, q_{i_m}\}$, то $\mathcal{L}(\mathcal{A})$ се описва с регулярния израз

$$\boldsymbol{r_{0,i_1}^{n}} + \boldsymbol{r_{0,i_2}^{n}} + \cdots + \boldsymbol{r_{0,i_m}^{n}}.$$

□

Примерни задачи

Задача 2.9. Приложете конструкцията от теоремата на Клини за да получите регулярен израз, който описва езика на автомата $\mathcal{A}$ изобразен на Фигура 2.20.



Фигура 2.20: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}[\![b^\star a(a+b)^\star]\!]$

Решение. Лесно се съобразява, че езикът на автомата от *Фигура 2.20* се описва с регулярния израз $b^\star a(a+b)^\star$. Следвайки означенията и конструкцията от доказателството на теоремата на Клини, езикът на този автомат се описва с регулярния израз $r^2_{0,1}$, защото началното състояние е q_0, финалното е q_1 и броят на състоянията в автомата е 2. Подробните сметки са следните:

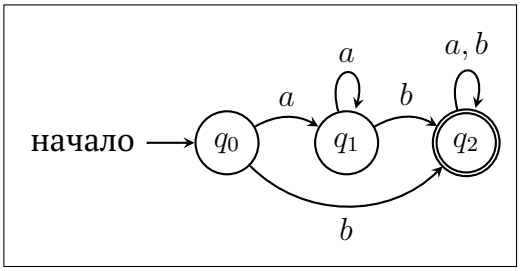
$$
\begin{aligned}
r^2_{0,1} &= \underbrace{r^1_{0,1}}_{b^\star a} + \underbrace{r^1_{0,1}}_{b^\star a} \cdot \underbrace{(r^1_{1,1})^\star}_{(\varepsilon+a+b)^\star} \cdot \underbrace{r^1_{1,1}}_{\varepsilon+a+b} && \text{// според (2.6)} \\
&= b^\star a + b^\star a(\varepsilon + a + b)^\star(\varepsilon + a + b) && \text{// просто заместваме} \\
&= b^\star a + b^\star a(\varepsilon + a + b)^+ && \text{// } r^+ \stackrel{\text{деф}}{=} r^\star \cdot r \\
&= b^\star a + b^\star a(a + b)^\star && \text{// } r^\star = (\varepsilon + r)^+ \\
&= b^\star a(\varepsilon + (a + b)^\star) && \text{// } r + rq = r(\varepsilon + q) \\
&= b^\star a(a + b)^\star. && \text{// } r^\star = \varepsilon + r^\star
\end{aligned}
$$

Тук използвахме, че:

$$
\begin{aligned}
r^1_{0,1} &= \underbrace{r^0_{0,1}}_{a} + \underbrace{r^0_{0,0}}_{\varepsilon+b} \cdot \underbrace{(r^0_{0,0})^\star}_{(\varepsilon+b)^\star} \cdot \underbrace{r^0_{0,1}}_{b} && \text{// според (2.6)} \\
&= a + (\varepsilon + b)(\varepsilon + b)^\star a && \text{// просто заместваме} \\
&= a + b^\star a && \text{// } r^\star = \varepsilon + r^\star \\
&= b^\star a \\
r^1_{1,1} &= \underbrace{r^0_{1,1}}_{\varepsilon+a+b} + \underbrace{r^0_{1,0}}_{\emptyset} \cdot \underbrace{(r^0_{0,0})^\star}_{(\varepsilon+b)^\star} \cdot \underbrace{r^0_{0,1}}_{a} && \text{// отново според (2.6)} \\
&= \varepsilon + a + b + \emptyset(\varepsilon + b)^\star a && \text{// просто заместваме} \\
&= \varepsilon + a + b. && \text{// защото } \emptyset \cdot r = \emptyset
\end{aligned}
$$

□

Задача 2.10. Приложете конструкцията от теоремата на Клини за да получите регулярен израз, който описва езика на автомата $\mathcal{A}$ изобразен на Фигура 2.21.



Фигура 2.21: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}[\![a^\star b(a+b)^\star]\!]$.

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

$$
\begin{aligned}
r^3_{0,2} &= \underbrace{r^2_{0,2}}_{a^\star b} + \underbrace{r^2_{0,2}}_{a^\star b} \cdot \underbrace{(r^2_{2,2})^\star}_{(\varepsilon+a+b)^\star} \cdot \underbrace{r^2_{2,2}}_{\varepsilon+a+b} && \text{// според (2.6)} \\
&= a^\star b + a^\star b \cdot (a+b)^\star \cdot (\varepsilon + a + b) && \text{// просто заместваме} \\
&= a^\star b + a^\star b \cdot (a+b)^\star && \text{// } r^\star = r^\star \cdot (\varepsilon + r) \\
&= a^\star b(\varepsilon + (a+b)^\star) && \text{// } r_1 + r_1 \cdot r_2 = r_1 \cdot (\varepsilon + r_2) \\
&= a^\star b(a+b)^\star. && \text{// } r^\star = \varepsilon + r^\star
\end{aligned}
$$

Тук използвахме, че:

$$
\begin{aligned}
r^2_{0,2} &= \underbrace{r^1_{0,2}}_{b} + \underbrace{r^1_{0,1}}_{a} \cdot \underbrace{(r^1_{1,1})^\star}_{a^\star} \cdot \underbrace{r^1_{1,2}}_{b} && \text{// според (2.6)} \\
&= b + a \cdot a^\star \cdot b && \text{// просто заместваме} \\
&= (\varepsilon + a^+) \cdot b && \text{// } r^+ \stackrel{\text{деф}}{=} r \cdot r^\star \\
&= a^\star b. && \text{// } r^\star = \varepsilon + r^+ \cdot r
\end{aligned}
$$

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

2.6 Автоматните езици са регулярни (алгебричен подход)

В този раздел ще опишем един алгебричен поглед към регулярните езици, с чиято помощ можем да построим регулярен израз по даден краен автомат.

Изглежда естествено да се запитаме дали можем да обобщим лемата на Ардън до системи от уравнения.

Теорема 2.4. Да разгледаме системата:

$$\left|\begin{array}{rcl} X_1 & = & L_{1,1}\cdot X_1 \cup L_{1,2}\cdot X_2 \cup \cdots \cup L_{1,n}\cdot X_n \cup M_1 \\ & \vdots & \\ X_n & = & L_{n,1}\cdot X_1 \cup L_{n,2}\cdot X_2 \cup \cdots \cup L_{n,n}\cdot X_n \cup M_n, \end{array}\right.$$

където $L_{i,j}$ и M_i са произволни езици, за $i,j = 1,\ldots,n$, като $\varepsilon \notin L_{i,j}$, за всяко $i,j = 1,\ldots,n$. Тогава тази система има *единствено* решение.

Доказателство. Индукция по броят на променливите n в системата.

Нека $n = 1$. Тогава системата представлява просто това:

$$X_1 = L_{1,1}\cdot X_1 \cup M_1.$$

Щом $\varepsilon \notin L_{1,1}$, от лемата на Ардън веднага следва, че тази система има *единствено* решение

$$S_1 = L_{1,1}^{\star}\cdot M_1.$$

Нека сега $n > 1$. Да разгледаме само последния ред на системата във вида, който ни трябва за да приложим лемата на Ардън.

$$X_n = L_{n,n}\cdot X_n \cup \bigcup_{i=1}^{n-1} L_{n,i}\cdot X_i \cup M_n.$$

Понеже $\varepsilon \notin L_{n,n}$, то за всеки избор на езици, с които да заменим променливите $X_1,\ldots,X_{n-1}$, горното уравнение ще има *единствено* решение и то е следното:

$$X_n = L_{n,n}^{\star}\cdot\Big(\bigcup_{i=1}^{n-1} L_{n,i}\cdot X_i \cup M_n\Big), \tag{2.7}$$

което, като разкрием скобите, придобива следния вид:

$$X_n = \bigcup_{i=1}^{n-1} L_{n,n}^{\star}\cdot L_{n,i}\cdot X_i \cup L_{n,n}^{\star}\cdot M_n.$$

Системата може да се запише по-компактно така:

$$\begin{array}{rl} X_1 = & \bigcup_{i=1}^{n} L_{1,i}\cdot X_i \cup M_1 \\ \vdots & \\ X_n = & \bigcup_{i=1}^{n} L_{n,i}\cdot X_i \cup M_n. \end{array}$$

Кратко описание на този подход може да се намери и в [23, стр. 134], [14, стр. 66].

Сега заместваме в първите $n-1$ реда на системата всяко срещане на X_n с неговото единствено решение, зависещо от $X_1, \ldots, X_{n-1}$. Получаваме следната система:

$$\left|\begin{array}{rcl} X_1 & = & \bigcup_{i=1}^{n-1}(L_{1,i} \cup L_{1,n} \cdot L_{n,n}^{\star} \cdot L_{n,i}) \cdot X_i \cup L_{1,n} \cdot L_{n,n}^{\star} \cdot M_n \cup M_1 \\ & \vdots & \\ X_{n-1} & = & \bigcup_{i=1}^{n-1}(L_{n-1,i} \cup L_{n-1,n} \cdot L_{n,n}^{\star} \cdot L_{n,i}) \cdot X_i \cup L_{n-1,n} \cdot L_{n,n}^{\star} \cdot M_n \cup M_{n-1}. \end{array}\right.$$

Това е вече система с $n-1$ уравнения и $n-1$ неизвестни. От условието знаем, че за всяко $k = 1, \ldots, n$ и всяко $i = 1, \ldots, n$, празната дума $\varepsilon \notin L_{k,i}$. Тогава е ясно, че за всяко $k = 1, \ldots, n-1$ и всяко $i = 1, \ldots, n-1$,

$$\varepsilon \notin L_{k,i} \cup L_{k,n} \cdot L_{n,n}^{\star} \cdot L_{n,i}.$$

От **(И.П.)** тази система има *единствено* решение съставено от езиците

$$S_1, \ldots, S_{n-1}.$$

Тогава уравнението (2.7) за X_n също има единствено решение

$$S_n = \bigcup_{i=1}^{n-1} L_{n,n}^{\star} \cdot L_{n,i} \cdot S_i \cup L_{n,n}^{\star} \cdot M_n.$$

Така получаваме, че $S_1, \ldots, S_n$ е единственото решение на първоначалната система. □

От доказателството на *Теорема 2.4* веднага се вижда, че ако се ограничим само до регулярни езици, то единственото решение на системата ще е също съставено от регулярни езици.

Следствие 2.1. Да разгледаме системата:

$$\left|\begin{array}{rcl} X_1 & = & L_{1,1} \cdot X_1 \cup L_{1,2} \cdot X_2 \cup \cdots \cup L_{1,n} \cdot X_n \cup M_1 \\ & \vdots & \\ X_n & = & L_{n,1} \cdot X_1 \cup L_{n,2} \cdot X_2 \cup \cdots \cup L_{n,n} \cdot X_n \cup M_n, \end{array}\right.$$

където $L_{i,j}$ и M_i са *регулярни* езици, за $i, j = 1, \ldots, n$, като $\varepsilon \notin L_{i,j}$, за всяко $i, j = 1, \ldots, n$. Тогава тази система има *единствено* решение, което е съставено от *регулярни* езици.

Следващата стъпка е да видим как системите от уравнения от регулярни изрази се „връзват" с автоматните езици. Да разгледаме произволен ДКА $\mathcal{A}$. Оказва се, че ние вече знаем как. Достатъчно е да си припомним *Твърдение 2.2*. Според него, езиците $\mathcal{L}_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ са решение на системата

$$\left|\begin{array}{rcl} X_1 &=& R_{1,1}\cdot X_1 \cup \cdots R_{1,n}\cdot X_n \cup P_1 \\ & \vdots & \\ X_n &=& R_{n,1}\cdot X_1 \cup \cdots R_{n,n}\cdot X_n \cup P_n. \end{array}\right.$$

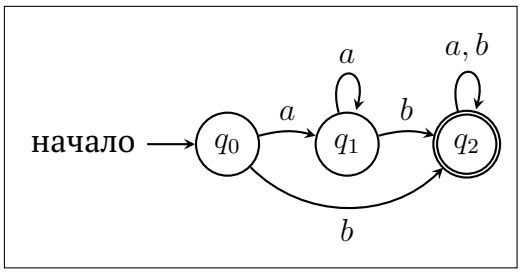
Знаем, че $R_{i,j} \stackrel{\text{деф}}{=} \{a \in \Sigma \mid \delta(q_i, a) = q_j\}$ е краен език и следователно е регулярен. Ясно е също така, че $\varepsilon \notin R_{i,j}$. Знаем също, че

$$P_i \stackrel{\text{деф}}{=} \begin{cases} \{\varepsilon\}, & \text{ако } q_i \in F \\ \emptyset, & \text{ако } q_i \notin F \end{cases},$$

които отново са крайни езици и следователно регулярни. Според *Следствие 2.1* системата има *единствено* решение, което е съставено от регулярни езици. Щом $\mathcal{L}_\mathcal{A}(q_1), \ldots, \mathcal{L}_\mathcal{A}(q_n)$ е решение на системата, то $\mathcal{L}_\mathcal{A}(q_1), \ldots, \mathcal{L}_\mathcal{A}(q_n)$ са регулярни езици. В частност, $\mathcal{L}(\mathcal{A})$ е регулярен език. Така доказахме следната теорема.

Теорема 2.5 (Клини [12])**.** Всеки автоматен език е регулярен.

Пример 2.7. Да разгледаме отново автомата от *Фигура 2.22*.



Фигура 2.22: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}[\![\boldsymbol{a^\star b(a+b)^\star}]\!]$.

На него съответства следната система от уравнения на езици:

$$\left|\begin{array}{rcl} \mathcal{L}_\mathcal{A}(q_0) &=& \{a\}\cdot\mathcal{L}_\mathcal{A}(q_1) \cup \{b\}\cdot\mathcal{L}_\mathcal{A}(q_2) \cup \emptyset \\ \mathcal{L}_\mathcal{A}(q_1) &=& \{a\}\cdot\mathcal{L}_\mathcal{A}(q_1) \cup \{b\}\cdot\mathcal{L}_\mathcal{A}(q_2) \cup \emptyset \\ \mathcal{L}_\mathcal{A}(q_2) &=& \{a\}\cdot\mathcal{L}_\mathcal{A}(q_2) \cup \{b\}\cdot\mathcal{L}_\mathcal{A}(q_2) \cup \{\varepsilon\}. \end{array}\right.$$

За простота, да преобразуваме системата в система от уравнения на регулярни изрази:

$$\left|\begin{array}{rcl} \boldsymbol{r_0} &=& \boldsymbol{a\cdot r_1 + b\cdot r_2 + \emptyset} \\ \boldsymbol{r_1} &=& \boldsymbol{a\cdot r_1 + b\cdot r_2 + \emptyset} \\ \boldsymbol{r_2} &=& \boldsymbol{a\cdot r_2 + b\cdot r_2 + \varepsilon}. \end{array}\right.$$

Въпреки, че $\emptyset$ нищо не добавя към езика, пишем $\emptyset$ за да видим общия вид на уравненията, които приличат на полиноми. Ясно е, че $\mathcal{L}[\![\boldsymbol{r_i}]\!] = \mathcal{L}_\mathcal{A}(q_i)$.

Да разгледаме само последния ред на системата:

$$r_2 = (a + b) \cdot r_2 + \varepsilon.$$

Чрез прилагане на лемата на Ардън получаваме, че единственото решение на това уравнение е $(a + b)^\star$. В първите два реда на системата заместваме r_2 с $(a + b)^\star$ и получаваме системата:

$$\left|\begin{array}{lcl} r_0 &=& a \cdot r_1 + b \cdot (a + b)^\star \\ r_1 &=& a \cdot r_1 + b \cdot (a + b)^\star \\ r_2 &=& (a + b)^\star. \end{array}\right.$$

Сега вече би трябвало да е ясно как продължаваме. Разглеждаме само втория ред на системата и прилагаме лемата на Ардън за него и получаваме, че единственото решение на втория ред от системата е $a^\star \cdot b \cdot (a + b)^\star$. Така получаваме, след заместване, следните уравнения:

$$\left|\begin{array}{lcl} r_0 &=& a \cdot a^\star \cdot b \cdot (a + b)^\star + b \cdot (a + b)^\star \\ r_1 &=& a^\star \cdot b \cdot (a + b)^\star \\ r_2 &=& (a + b)^\star. \end{array}\right.$$

Тук е сравнително лесно да опростим регулярния израз, но в общия случай това може да е много трудно. Оказва се, че $r_0 = r_1$. По-късно ще видим, че това означава, че можем да „слеем" двете състояния и да получим по-компактен автомат за същия език.

Заключаваме, че езикът на автомата $\mathcal{A}$ може да се опише с регулярния израз:

$$\begin{aligned} r_0 &= a \cdot a^\star \cdot b \cdot (a + b)^\star + b \cdot (a + b)^\star \\ &= (a \cdot a^\star + \varepsilon) \cdot b \cdot (a + b)^\star && \text{// } r \cdot p + p = (r + \varepsilon) \cdot p \\ &= a^\star \cdot b \cdot (a + b)^\star. && \text{// } a \cdot a^\star + \varepsilon = a^\star \end{aligned}$$

Макар и нашият подход тук да беше алгебричен, от горния пример лесно се вижда как може да се построи алгоритъм за намиране на регулярен израз описващ езика на даден ДКА.

Твърдение 2.9. Съществува алгоритъм, за който при вход краен детерминиран автомат $\mathcal{A}$, извежда като изход регулярен израз r, такъв че $\mathcal{L}(\mathcal{A}) = \mathcal{L}[\![r]\!]$.

За подробно изложение на този въпрос, вижте [25, стр. 69]. Алгоритъмът на практика следва идеята изложена в Пример 2.7.

2.7 Каноничен автомат

Нека въведем следната операция за произволна буква a,

Може би по-добро означение е L_a вместо $a^{-1}(L)$.

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

Нека още тук да въведем следното означение, което ще ни бъде полезно по-нататък:

Бжозовски [5] описва алгоритъм за строене на автомат по регулярен израз [23, стр. 112].

$$\varepsilon(L) \stackrel{\text{деф}}{=} \begin{cases} \{\varepsilon\}, & \text{ако } \varepsilon \in L \\ \emptyset, & \text{иначе.} \end{cases}$$

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

Задача 2.11. 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) $a^{-1}(L_1 \cdot L_2) = a^{-1}(L_1) \cdot L_2 \cup \varepsilon(L_1) \cdot a^{-1}(L_2)$;

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

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

Аналогично, за произволна дума α, нека положим

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

Оттук е ясно, че

$$L = \{\omega \in \Sigma^\star \mid \varepsilon \in \omega^{-1}(L)\}. \tag{2.8}$$

Твърдение 2.10. 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:

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

□

Нека е даден произволен език L. Ще покажем конструкция на детерминиран автомат

$$\mathcal{B}_L = \langle \Sigma, Q_L, q^L_{\texttt{start}}, \delta_L, F_L \rangle,$$

който разпознава L. Този автомат ще наречем **автомат на Бжозовски**. Ще дивим, че ако L е регулярен, то $\mathcal{B}_L$ ще бъде детерминиран краен автомат, но ако L не е регулярен, то $\mathcal{B}_L$ ще бъде детерминиран *безкраен* автомат. Конструкцията на автомата $\mathcal{B}_L$ е следната:

Тук виждате, че езиците играят ролята на състояния на автомата на Бжозовски [23, стр. 112]. На автоматът на Бжозовски може да се гледа като на *каноничния* автомат за дадения език [8, стр. 53].

- Състоянията Q_L на автомата на Бжозовски ще бъдат езици над азбуката Σ, където:
$$Q_L \stackrel{\text{деф}}{=} \{\alpha^{-1}(L) \mid \alpha \in \Sigma^\star\}.$$
- Началното състояние $q_{\texttt{start}}$ на автомата на Бжозовски е дефинирано като:
$$q^L_{\texttt{start}} \stackrel{\text{деф}}{=} L \qquad // \; L = \varepsilon^{-1}(L).$$
- За произволно състояние (език) $M \in Q_L$ и буква a, дефинираме функцията на преходите ето така:
$$\delta_L(M, a) \stackrel{\text{деф}}{=} a^{-1}(M).$$
- Финалните състояния на автомата на Бжозовски са следните:
$$F_L \stackrel{\text{деф}}{=} \{M \in Q_L \mid \varepsilon \in M\}.$$

Твърдение 2.11. За всяка дума α и всяко състояние $M \in Q_L$ е изпълнено равенството:

$$\delta^\star_L(M, \alpha) = \alpha^{-1}(M). \tag{2.9}$$

✍ Опитайте се да докажете това твърдение сами!

Упътване. Индукция по дължината на думата α, като използвате, че

$$(\alpha b)^{-1}(M) = b^{-1}(\alpha^{-1}(M)).$$

□

Доказателство.

- Твърдението очевидно е изпълнено за $\alpha = \varepsilon$, защото

$$\delta_L^\star(M, \varepsilon) \overset{\text{деф}}{=} M = \varepsilon^{-1}(M).$$

- Да примем, че за думи α с дължина n е изпълнено Свойство (2.9).
- Сега да разгледаме дума α с дължина $n+1$, т.е. $\alpha = \beta a$ и $|\beta| = n$. Тогава

$$\begin{aligned}
\delta_L^\star(M, \beta a) &= \delta_L(\delta_L^\star(M, \beta), a) && \text{// от деф. на } \delta_L^\star \\
&= \delta_L(\beta^{-1}(M), a) && \text{// от } \textbf{(И.П.)} \\
&= a^{-1}(\beta^{-1}(M)) && \text{// от деф. на } \delta_L \\
&= (\beta a)^{-1}(M). && \text{// от } \textit{Твърдение 2.10}
\end{aligned}$$

□

Твърдение 2.12. За произволен език L, $L = \mathcal{L}(\mathcal{B}_L)$.

Упътване. Съобразете, че имаме следните еквивалентности:

$$\begin{aligned}
\alpha \in L &\Leftrightarrow \varepsilon \in \alpha^{-1}(L) && \text{// от } \textit{Задача 2.8} \\
&\Leftrightarrow \varepsilon \in \delta_L^\star(L, \alpha) && \text{// от } \textit{Твърдение 2.11} \\
&\Leftrightarrow \delta_L^\star(q^L_{\texttt{start}}, \alpha) \in F_L && \text{// } \varepsilon \in \alpha^{-1}(L) \\
&\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{B}_L). && \text{// деф. на език на автомат}
\end{aligned}$$

□

Тук е важно да отбележим, че все още не знаем дали $\mathcal{B}_L$ има крайно много състояния. В *Пример 2.8* ще видим един детерминиран безкраен автомат за език, който не е регулярен.

Примерни задачи

Задача 2.12. Постройте автомат $\mathcal{B}$ по метода на Бжозовски за регулярния език

$$L = \mathcal{L}[\![((\boldsymbol{a} + \boldsymbol{b})^+ \cdot \boldsymbol{a})^\star]\!].$$

Решение.

Да напомним, че $\boldsymbol{r}^+ \stackrel{\text{деф}}{=} \boldsymbol{r} \cdot \boldsymbol{r}^\star$. Очевидно е, че $\boldsymbol{r}^\star = \boldsymbol{\varepsilon} + \boldsymbol{r}^+$. Веднага се вижда, че L ще бъде и финално състояние, защото $\varepsilon \in L$. Често ще използваме съкратения запис αL вместо $\{\alpha\} \cdot L$.

- Ясно е, че ще започнем от началното състояние L. Удобно е да имаме предвид следното представяне, при което явно да се вижда кои думи на L започват с буквата a и кои с буквата b:

$$\begin{aligned} L &= (\{a,b\}^+ \cdot \{a\})^\star \\ &= \{\varepsilon\} \cup (\{a,b\}^+ \cdot \{a\})^+ \\ &= \{\varepsilon\} \cup \{a,b\}^+ \cdot \{a\} \cdot L \\ &= \{\varepsilon\} \cup \{a,b\} \cdot \{a,b\}^\star aL \\ &= \{\varepsilon\} \cup a\{a,b\}^\star aL \cup b\{a,b\}^\star aL \end{aligned}$$

- 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 M и

$$\delta(L,a) \stackrel{\text{деф}}{=} M \quad \text{// защото } a^{-1}(L) = M$$
$$\delta(L,b) \stackrel{\text{деф}}{=} M. \quad \text{// защото } b^{-1}(L) = M$$

Веднага се вижда, че M няма да бъде финално състояние.

- 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

$$a^{-1}(M) = L \cup M$$
$$b^{-1}(M) = M.$$

Обърнете внимание, че $L \subset N$, но L и N са различни състояния.

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 N и тогава

$$\delta(M,a) \stackrel{\text{деф}}{=} N \quad \text{// защото } a^{-1}(M) = N$$
$$\delta(M,b) \stackrel{\text{деф}}{=} M \quad \text{// защото } b^{-1}(M) = M.$$

- Да разгледаме следното представяне:

$$\begin{aligned} N &= L \cup M \\ &= \{\varepsilon\} \cup aM \cup bM \cup M \\ &= \{\varepsilon\} \cup aM \cup bM \cup aL \\ &= \{\varepsilon\} \cup aN \cup bM. \end{aligned}$$

Веднага можем да съобразим, че

$$a^{-1}(N) = N$$
$$b^{-1}(N) = M.$$

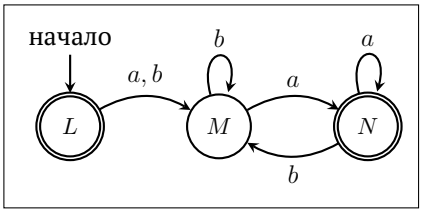
Сега полагаме:

$$\delta(N,a) \stackrel{\text{деф}}{=} N \quad \text{// защото } a^{-1}(N) = N$$
$$\delta(N,b) \stackrel{\text{деф}}{=} M \quad \text{// защото } b^{-1}(N) = M.$$

- Нямаме повече нови състояния. Следователно, $Q \stackrel{\text{деф}}{=} \{L, M, N\}$.
- Понеже $\varepsilon \in L$ и $\varepsilon \in N$ е ясно, че

$$F = \{L, N\}.$$

Сега вече сме готови да нарисуваме картинка на автомата.



Фигура 2.23: Автомат на Бжозовски за езика L.

□

Задача 2.13. 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 L, за което е ясно, че няма да бъде финално, защото $\varepsilon \notin L$. Първата стъпка е лесна, защото е явно, че всички думи на L започват с буквата a.
- $a^{-1}(L) = \{a,b\}^\star b \stackrel{\text{деф}}{=} M$.
  Имаме, че $M \neq L$, защото $b \in M$, но $b \notin L$. Тогава

  $$\delta(L,a) \stackrel{\text{деф}}{=} M \quad \text{// защото } a^{-1}(L) = M$$
  $$\delta(L,b) \stackrel{\text{деф}}{=} \emptyset \quad \text{// защото } b^{-1}(L) = \emptyset$$

  $\emptyset$ е съвсем нормален език и напълно нормално да имаме състояние $\emptyset$.
- За по-нататък ще е удобно да представим M по следния начин, така че да се вижда кои думи на M започват с буквата a и кои с буквата b:

  $$\begin{aligned} M &= \{a,b\}^\star b \\ &= \{a,b\}^+ b \cup \{b\} \\ &= a \cdot \{a,b\}^\star \cdot b \cup b \cdot \{a,b\}^\star \cdot b \cup \{b\} \\ &= aM \cup bM \cup \{b\}. \end{aligned}$$

  Сега е ясно, че

  $$a^{-1}(M) = M$$
  $$b^{-1}(M) = \{\varepsilon\} \cup M.$$

  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

  $$\delta(M,a) \stackrel{\text{деф}}{=} M \quad \text{// защото } a^{-1}(M) = M$$
  $$\delta(M,b) \stackrel{\text{деф}}{=} N \quad \text{// защото } b^{-1}(M) = N$$

- Можем да представим езика N по следния начин:

  $$N = \{\varepsilon\} \cup aM \cup bM \cup \{b\}.$$

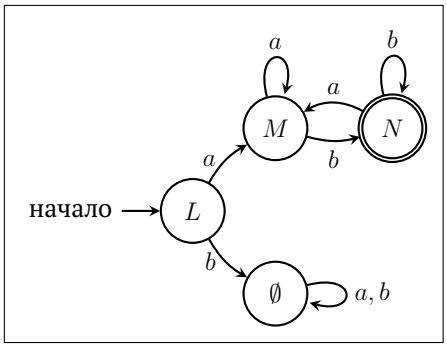
  Тогава имаме, че:

  $$\delta(N,a) \stackrel{\text{деф}}{=} M \quad \text{// защото } a^{-1}(N) = M$$
  $$\delta(N,b) \stackrel{\text{деф}}{=} N \quad \text{// защото } b^{-1}(N) = M.$$

- Завършваме с дефиницията на функцията на преходите като:

  $$\delta(\emptyset,a) \stackrel{\text{деф}}{=} \emptyset \quad \text{// защото } a^{-1}(\emptyset) = \emptyset$$
  $$\delta(\emptyset,b) \stackrel{\text{деф}}{=} \emptyset \quad \text{// защото } b^{-1}(\emptyset) = \emptyset.$$

- Съобразете сами, че $F = \{N\}$.



Фигура 2.24: Автомат за $\mathcal{L}[\![\boldsymbol{a} \cdot (\boldsymbol{a} + \boldsymbol{b})^\star \cdot \boldsymbol{b}]\!]$ по метода на Бжозовски.

□

Да припомним, че в *Задача 2.4* се искаше да се докаже, че същият език е регулярен. Ние направихме това като построихме автомат за L и доказахме, че той разпознава L.

Задача 2.14. Постройте автомат $\mathcal{B}$ по метода на Бжозовски, който разпознава езика

$$L = \{\alpha \in \{0,1\}^\star \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\}.$$

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 L на автомата на Бжозовски не е финално, защото $\varepsilon \notin L$. Да започнем с преходите от началното състояние:

$$\begin{aligned}
0^{-1}(L) &= \{\alpha \mid 0\alpha \in L\} \\
&= \{\alpha \mid \overline{0\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= L. \\
1^{-1}(L) &= \{\alpha \mid 1\alpha \in L\} \\
&= \{\alpha \mid \overline{1\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&\overset{\text{деф}}{=} M.
\end{aligned}$$

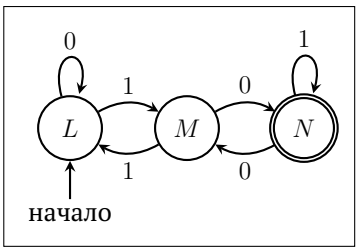
Лесно се съобразява, че езикът M е различен от L, защото например думата $10 \in L$, но $10 \notin M$. Също така, понеже $2^{|\varepsilon|} + \overline{\varepsilon}_{(2)} = 1$, то $\varepsilon \notin M$ и следователно M няма да бъде финално състояние в автомата на Бжозовски. Продължаваме нататък:

$$\begin{aligned}
0^{-1}(M) &= \{\alpha \mid 0\alpha \in M\} \\
&= \{\alpha \mid 2^{|0\alpha|} + \overline{0\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&\overset{\text{деф}}{=} N.
\end{aligned}$$

$$\begin{aligned}
1^{-1}(M) &= \{\alpha \mid 1\alpha \in M\} \\
&= \{\alpha \mid 2^{|1\alpha|} + \overline{1\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 2 \cdot 2^{|\alpha|} + 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid 3 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha \mid \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= L.
\end{aligned}$$

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 N ще бъде финално състояние в автомата на Бжозовски, защото $\varepsilon \in N$. Продължаваме нататък с преходите от 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 2.25: Автомат на Бжозовски за езика L.

□

Задача 2.15. Да фиксираме азбуката $\Sigma = \{a, b\}$. Постройте автомат по метода на Бжозовски за регулярния език

$$L = \{\omega \in \Sigma^\star \mid |\omega|_a \text{ е четно и } |\omega|_b = 1\}.$$

Решение.

Вече сме достатъчно опитни за да ни е очевидно, че L е регулярен. Да напомним, че $|\omega|_a$ означава броя на срещанията на a в думата ω.

Да започнем с преходите от началното състояние L:

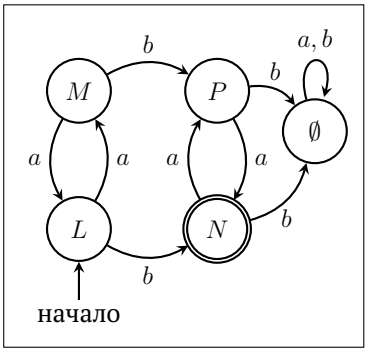
$$\begin{aligned} a^{-1}(L) &= \{\omega \in \Sigma^\star \mid |\omega|_a \text{ е нечет. и } |\omega|_b = 1\} \\ &\stackrel{\text{деф}}{=} M \\ b^{-1}(L) &= \{\omega \in \Sigma^\star \mid |\omega|_a \text{ е четно и } |\omega|_b = 0\} \\ &\stackrel{\text{деф}}{=} N. \end{aligned}$$

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 M:

$$\begin{aligned} a^{-1}(M) &= L; \\ b^{-1}(M) &= \{\omega \in \Sigma^\star \mid |\omega|_a \text{ е нечет. и } |\omega|_b = 0\} \\ &\stackrel{\text{деф}}{=} P. \end{aligned}$$

Ясно е, че $P \neq M, L, N$, защото $a \in P$, но $a \notin M, L, N$. Завършваме с преходите от състоянията N и P:

$$\begin{aligned} a^{-1}(N) &= P \quad \text{и} \quad b^{-1}(N) = \emptyset \\ a^{-1}(P) &= N \quad \text{и} \quad b^{-1}(P) = \emptyset. \end{aligned}$$



Фигура 2.26: 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 [23, стр. 113].

Пример 2.8. Да разгледаме езика $L = \{a^n b^n \mid n \in \mathbb{N}\}$. Ние вече знаем от *Пример 2.9*, че L не е регулярен език. Да се опитаме да построим автомат, който го разпознава.

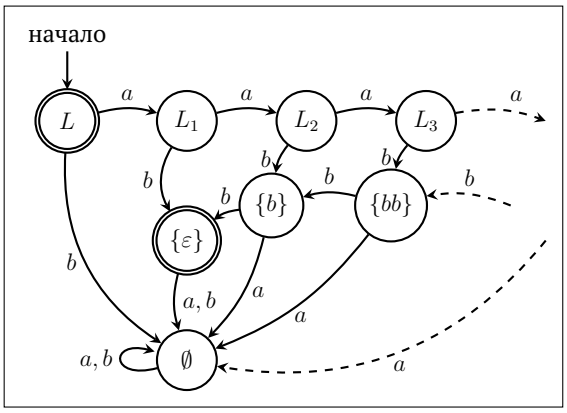
Нека за всяко k да разгледаме езика

$$L_k \stackrel{\text{деф}}{=} \{a^n b^{n+k} \mid n \in \mathbb{N}\}.$$

Да видим какво се получава като приложим процедурата за строене на минимален автомат.

- $a^{-1}(L) = L_1$ и $b^{-1}(L) = \emptyset$;
- $a^{-1}(L_1) = L_2$ и $b^{-1}(L_1) = \{\varepsilon\}$;
- $a^{-1}(\{\varepsilon\}) = b^{-1}(\{\varepsilon\}) = \emptyset$;
- 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 2.27: *Безкраен* автомат за езика $\{a^n b^n \mid n \in \mathbb{N}\}$ построен по метода на Бжозовски.

2.8 Регулярните езици са автоматни (алгебричен подход)

За следващата теорема ще се наложи да работим с множества от езици. Нека $\mathcal{X}$ е множество от езици и L е език. Да обобщим операцията конкатенация така:

$$L \cdot \mathcal{X} \stackrel{\text{деф}}{=} \{L \cdot M \mid M \in \mathcal{X}\}$$
$$\mathcal{X} \cdot L \stackrel{\text{деф}}{=} \{M \cdot L \mid M \in \mathcal{X}\}.$$

Лема 2.4. Ако L е регулярен език, то Q_L е крайно множество.

[23, стр. 142]. Доказателството на тази лема не е лесно, но по този начин можем да си спестим разглеждането на конструкциите от *Раздел 2.4*.

Доказателство. Индукция по построението на регулярните езици.

- Нека $L = \emptyset$. Ясно е, че $Q_L = \{\emptyset\}$.
- Нека $L = \{\varepsilon\}$. Ясно е, че $Q_L = \{\{\varepsilon\}, \emptyset\}$.
- Нека $L = \{a\}$. Ясно е, че $Q_L = \{\{a\}, \{\varepsilon\}, \emptyset\}$.
- Нека $L = L_1 \cup L_2$. Ясно е, че за произволна дума ω,

  $$\omega^{-1}(L_1 \cup L_2) = \omega^{-1}(L_1) \cup \omega^{-1}(L_2).$$

  Това означава, че за всяка дума ω, съществуват езици $M_1 \in Q_{L_1}$ и $M_2 \in Q_{L_2}$, за които $\omega^{-1}(L_1 \cup L_2) = M_1 \cup M_2$. Да положим

  $$\mathcal{X} \stackrel{\text{деф}}{=} \{M_1 \cup M_2 \mid M_1 \in Q_{L_1} \,\&\, M_2 \in Q_{L_2}\}.$$

  Тогава:

  $$Q_{L_1 \cup L_2} \subseteq \mathcal{X}.$$

  Да разгледаме функцията $f : Q_{L_1} \times Q_{L_2} \to \mathcal{X}$, където

  $$f(M_1, M_2) \stackrel{\text{деф}}{=} M_1 \cup M_2.$$

  Ясно е, че f е сюрекция, откъдето следва, че $|\mathcal{X}| \leq |Q_{L_1} \times Q_{L_2}|$. Оттук заключаваме, че:

  $$|Q_{L_1 \cup L_2}| \leq |\mathcal{X}| \leq |Q_{L_1} \times Q_{L_2}| = |Q_{L_1}| \cdot |Q_{L_2}|.$$

  От **(И.П.)** имаме, че Q_{L_1} и Q_{L_2} са крайни множества. Следователно $Q_{L_1 \cup L_2}$ също е крайно множество.

- Нека $L = L_1 \cdot L_2$. Тогава, за произволна дума ω,

$$\omega^{-1}(L_1 \cdot L_2) = \omega^{-1}(L_1) \cdot L_2 \cup \bigcup_{\omega=\omega_1\cdot\omega_2} \varepsilon(\omega_1^{-1}(L_1)) \cdot \omega_2^{-1}(L_2).$$

  Това означава, че за всяка дума ω, съществува език $M \in Q_{L_1} \cdot L_2$ и множество от езици $\mathcal{X} \subseteq Q_{L_2}$, за които

$$\omega^{-1}(L_1 \cdot L_2) = M \cup \bigcup \mathcal{X}.$$

  Така получаваме, че

$$Q_{L_1\cdot L_2} \subseteq \left\{ M \cup \bigcup \mathcal{X} \mid M \in Q_{L_1} \cdot L_2 \;\&\; \mathcal{X} \in \mathcal{P}(Q_{L_2}) \right\}.$$

  Понеже $|Q_{L_1} \cdot L_2| \leq |Q_{L_1}|$, то можем да заключим, че:

$$|Q_{L_1\cdot L_2}| \leq |Q_{L_1} \times \mathcal{P}(Q_{L_2})| = |Q_{L_1}| \cdot 2^{|Q_{L_2}|}.$$

  От **(И.П.)** имаме, че Q_{L_1} и Q_{L_2} са крайни множества. Следователно $Q_{L_1\cdot L_2}$ също е крайно множество.

- Използвайки същата идея както по-горе, получваме следното:

$$\omega^{-1}(L^{n+1}) = \bigcup_{\ell+r=n} \bigcup_{\omega=\omega_1\cdot\omega_2} \varepsilon(\omega_1^{-1}(L^\ell)) \cdot \omega_2^{-1}(L) \cdot L^r.$$

- Сега използваме, че $L^\star = \bigcup_n L^n$ и тогава:

$$\begin{aligned} \omega^{-1}(L^\star) &= \omega^{-1}(\bigcup_n L^n) \\ &= \bigcup_n \omega^{-1}(L^n). \end{aligned}$$

  Заключаваме, че:

$$\begin{aligned} \omega^{-1}(L^\star) &= \bigcup_\ell \bigcup_r \bigcup_{\omega_1\cdot\omega_2=\omega} \varepsilon(\omega_1^{-1}(L^\ell)) \cdot \omega_2^{-1}(L) \cdot L^r \\ &= \bigcup_\ell \bigcup_{\omega_1\cdot\omega_2=\omega} \varepsilon(\omega_1^{-1}(L^\ell)) \cdot \omega_2^{-1}(L) \cdot L^\star. \end{aligned}$$

  Така получаваме, че за всяко ω, съществува множество от езици

$$\mathcal{X} \subseteq Q_L \cdot L^\star,$$

  за което $\omega^{-1}(L^\star) = \bigcup \mathcal{X}$. Така получаваме, че

$$Q_{L^\star} \subseteq \left\{ \bigcup \mathcal{X} \mid \mathcal{X} \in \mathcal{P}(Q_L \cdot L^\star) \right\}.$$

  Понеже $|Q_L \cdot L^\star| \leq |Q_L|$, то можем да заключим, че:

$$|Q_{L^\star}| \leq |\mathcal{P}(Q_L \cdot L^\star)| \leq 2^{|Q_L|}.$$

  От **(И.П.)** имаме, че Q_L е крайно множество. Следователно $Q_{L^\star}$ също е крайно множество.

□

Теорема 2.6 (Клини[12])**.** Всеки регулярен език е автоматен.

Алтернативно доказателство на тази теорема е дадено в *Раздел 2.4*.

Доказателство. Знаем от *Твърдение 2.12*, че $\mathcal{L}(\mathcal{B}_L) = L$. Щом L е регулярен език, то от *Лема 2.4* имаме, че $\mathcal{B}_L$ е ДКА. □

Следствие 2.2 (Критерий за регулярност на език)**.** Един език L е регулярен точно тогава, когато $\mathcal{B}_L$ има крайно много състояния.

Доказателство. Първо, ако L е регулярен, то от *Лема 2.4* веднага имаме, че $\mathcal{B}_L$ е ДКА.

Второ, ако $\mathcal{B}_L$ има крайно много състояния, то $\mathcal{B}_L$ е ДКА. Понеже $\mathcal{L}(\mathcal{B}_L) = L$ според *Твърдение 2.12*, то L е автоматен и следователно регулярен по теоремата на Клини, за която имаме доказателство с алгоритмичен подход и алгебричен подход. □

Използвайки подобни разсъждения, можем директно да съобразим, че регулярните езици са затворени и относно операциите сечение и допълнение.

Следствие 2.3. Ако L_1 и L_2 са регулярни езици, то $L_1 \cap L_2$ и $L_1 \setminus L_2$ са регулярни езици.

Сравнете с *Твърдение 2.3*

Доказателство. Първо, съобразете, че $\omega^{-1}(L_1 \cap L_2) = \omega^{-1}(L_1) \cap \omega^{-1}(L_2)$. Това означава, че

$$Q_{L_1 \cap L_2} \subseteq \{M_1 \cap M_2 \mid M_1 \in Q_{L_1} \ \& \ M_2 \in Q_{L_2}\}.$$

Следователно,

$$|Q_{L_1 \cap L_2}| \leq |Q_{L_1} \times Q_{L_2}| = |Q_{L_1}| \cdot |Q_{L_2}|.$$

Второ, съобразете, че $\omega^{-1}(L_1 \setminus L_2) = \omega^{-1}(L_1) \setminus \omega^{-1}(L_2)$. Оттук, отново получаваме, че

$$Q_{L_1 \setminus L_2} \subseteq \{M_1 \setminus M_2 \mid M_1 \in Q_{L_1} \ \& \ M_2 \in Q_{L_2}\}.$$

Следователно,

$$|Q_{L_1 \setminus L_2}| \leq |Q_{L_1} \times Q_{L_2}| = |Q_{L_1}| \cdot |Q_{L_2}|.$$

□

Джон Конуей [6, стр. 43] предлага да обобщим операцията $\omega^{-1}(L)$ по следния начин. Нека M и L са произволни езици. Да положим

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

Задача 2.16. Докажете, че:

- $M^{-1}(\emptyset) = \emptyset$;
- $M^{-1}(\{a\}) = \begin{cases} \emptyset, & \text{ако } a \notin M \\ \{\varepsilon\}, & \text{ако } a \in M \\ \{a\}, & \text{ако } \varepsilon \in M \ \& \ a \notin M \\ \{\varepsilon, a\}, & \text{ако } \varepsilon \in M \ \& \ a \in M. \end{cases}$
- $M^{-1}(L_1 \cup L_2) = M^{-1}(L_1) \cup M^{-1}(L_2)$.
- $M^{-1}(L_1 \cdot L_2) = M^{-1}(L_1) \cdot L_2 \cup (L_1^{-1}(M))^{-1}(L_2)$;
- $M^{-1}(L^\star) = \varepsilon(M) \cup ((L^\star)^{-1}(M))^{-1}(L) \cdot L^\star$;
- $(M_1 \cup M_2)^{-1}(L) = M_1^{-1}(L) \cup M_2^{-1}(L)$;
- $(M_1 \cdot M_2)^{-1}(L) = M_2^{-1}(M_1^{-1}(L))$.

Задача 2.17. Докажете, че ако L е регулярен, а M е произволен език, то $M^{-1}(L)$ е регулярен.

2.9 Критерий за регулярност (автоматен подход)

Лема 2.5 (Лема за покачването). Нека L да бъде *безкраен* регулярен език. Съществува число $p \geq 1$, зависещо само от L, за което за всяка дума $\alpha \in L$, за която $|\alpha| \geq p$, може да бъде записана във вида $\alpha = xyz$ и

1) $|y| \geq 1$;

2) $|xy| \leq p$;

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

На англ. се нарича *Pumping Lemma*, макар според някои студенти да се нарича *Pumpkin Lemma*. Има подобна лема и за безконтекстни езици, която ще разгледаме по-нататък. Обърнете внимание, че $0 \in \mathbb{N}$ и $xy^0z = xz$. Тази лема я има на практика във всеки добър учебник по този предмет. Например, [18, стр. 88], [25, стр. 77], [10, стр. 55]. Оригиналното доказателство е на Бар-Хилел, Перлес и Шамир [3].

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

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

е краен детерминиран автомат, за който $L = \mathcal{L}(\mathcal{A})$. Да положим $p = |Q|$ и нека $\alpha = a_1 a_2 \cdots a_k$ е дума, за която $k \geq p$. Да разгледаме изпълнението на α върху $\mathcal{A}$:

$$q_{\mathtt{start}} \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_p} q_p \cdots \xrightarrow{a_k} q_k$$

и по специално първите p стъпки. Понеже в това доказателство ще работим с индексите на състоянията, нека за улеснение да положим $q_0 = q_{\mathtt{start}}$. Да положим $p = |Q|$ и нека $\alpha = a_1 a_2 \cdots a_k$ е дума, за която $k \geq p$. Тъй като $|Q| = p$, а по пътя

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_p} q_p$$

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 $|xy| = j \leq p$. Можем да опишем изчислението по следния начин:

$$(q_0, xyz) \vdash_{\mathcal{A}}^{\star} (q_i, yz) \vdash_{\mathcal{A}}^{\star} (q_j, z) \vdash_{\mathcal{A}}^{\star} (q_k, \varepsilon).$$

Да разгледаме случая за $n = 0$. Думата $xy^0z = xz \in L$, защото имаме следното изчисление:

$$q_0 \underbrace{\xrightarrow{a_1} q_1 \cdots \xrightarrow{a_i}} q_i \underbrace{\xrightarrow{a_{j+1}} q_{j+1} \cdots \xrightarrow{a_k}} q_k \in F,$$

защото $q_i = q_j$, или с други думи,

$$(q_0, xz) \vdash^\star_{\mathcal{A}} (q_i, z) \vdash^\star_{\mathcal{A}} (q_k, \varepsilon).$$

Сега да разгледаме случая $n = 2$. Тогава думата $xy^2z = xyyz \in L$, защото имаме следното изчисление:

$$q_0 \underbrace{\xrightarrow{a_1} q_1 \cdots \xrightarrow{a_i}}_{x} q_i \underbrace{\xrightarrow{a_{i+1}} q_{i+1} \cdots \xrightarrow{a_j}}_{y} \underbrace{q_j}_{q_i} \underbrace{\xrightarrow{a_{i+1}} q_{i+1} \cdots \xrightarrow{a_j}}_{y} \underbrace{q_j}_{q_i} \underbrace{\xrightarrow{a_{j+1}} \cdots \xrightarrow{a_k}}_{z} q_k \in F,$$

или с други думи,

$$(q_0, xyyz) \vdash^\star_{\mathcal{A}} (q_i, yyz) \vdash^\star_{\mathcal{A}} (q_i, yz) \vdash^\star_{\mathcal{A}} (q_i, z) \vdash^\star_{\mathcal{A}} (q_k, \varepsilon).$$

Аналогично, за произволно $n \geq 2$, имаме изчислението:

$$q_0 \underbrace{\xrightarrow{a_1} q_1 \cdots \xrightarrow{a_i}}_{x} \underbrace{q_i \underbrace{\xrightarrow{a_{i+1}} q_{i+1} \cdots \xrightarrow{a_j}}_{y} q_i \underbrace{\xrightarrow{a_{i+1}} q_{i+1} \cdots \xrightarrow{a_j}}_{y} q_i \cdots q_i}_{n \text{ пъти}} \underbrace{\xrightarrow{a_{j+1}} \cdots \xrightarrow{a_k}}_{z} q_k,$$

или с други думи,

$$(q_0, xy^nz) \vdash^\star_{\mathcal{A}} \underbrace{(q_i, y^nz) \vdash^\star_{\mathcal{A}} (q_i, y^{n-1}z) \vdash^\star_{\mathcal{A}} \cdots \vdash^\star_{\mathcal{A}} (q_i, z)}_{n \text{ пъти}} \vdash^\star_{\mathcal{A}} (q_k, \varepsilon).$$

От направените по-горе разсъждения, можем да заключим, че за всяко естествено число n е изпълнено, че $xy^nz \in L$. □

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

Следствие 2.4 (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.

Подобно е изложението и в [13, стр. 70].

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

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

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

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

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 $\neg\exists\forall\exists\forall(\dots) \equiv \forall\exists\forall\exists\neg(\dots)$

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

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

Използваме, че

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

Горната формула е еквивалентна на:

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

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 $xy^i z \notin L$,

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

Пример 2.9. Да видим защо езикът $L = \{a^n b^n \mid n \in \mathbb{N}\}$ 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 $\alpha = a^p b^p \in L$. 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 $xy^i z \notin L$. Понеже $|xy| \leq p$, то $y = a^k$, за $1 \leq k \leq p$. Тогава ако вземем $i = 0$, получаваме $xy^0 z = a^{p-k} b^p$. 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 2.4* 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*:

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

Приложения

[11, стр. 93].

В този раздел ще видим, че с помощта на лемата за покачването някои основни проблеми са алгоритмично разрешими за автоматни езици. По-нататък ще разгледаме тези проблеми и за други видове езици и ще видим, че не винаги те са алгоритмично разрешими.

Проблемът за празнота: Съществува ли алгоритъм, който по вход краен автомат $\mathcal{A}$, определя дали $\mathcal{L}(\mathcal{A}) = \emptyset$?

Проблемът за включване: Съществува ли алгоритъм, който по вход два крайни автомата $\mathcal{A}$ и $\mathcal{B}$, определя дали $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$?

Проблемът за равенство: Съществува ли алгоритъм, който по вход два крайни автомата $\mathcal{A}$ и $\mathcal{B}$, определя дали $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$?

Проблемът за безкрайност: Съществува ли алгоритъм, който по вход краен автомат $\mathcal{A}$, определя дали $|\mathcal{L}(\mathcal{A})| = \infty$?

Проблемът за универсалност: Съществува ли алгоритъм, който по вход краен автомат $\mathcal{A}$, определя дали $\mathcal{L}(\mathcal{A}) = \Sigma^\star$?

Вече имаме всичко необходимо за да отговорим сравнително лесно на тези въпроси. Първо ще разгледаме две помощни твърдения, които на практика следват директно от лемата за покачването.

Твърдение 2.13. Нека $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ е произволен ДКА. Тогава следните условия са еквивалентни:

(1) $\mathcal{L}(\mathcal{A})$ е *непразен*;

(2) $\mathcal{A}$ разпознава дума α, за която $|\alpha| < |Q|$.

Доказателство. Да положим $L = \mathcal{L}(\mathcal{A})$. Ще разгледаме двете посоки на твърдението.

$(1) \Rightarrow (2)$**.** 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 лемата за покачването, 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 $|\alpha| < |Q|$.

$(2) \Rightarrow (1)$**.** Тази посока е тривиална. Ако L съдържа дума α, за която $|\alpha| < |Q|$, то е очевидно, че L е непразен език.

□

Твърдение 2.14. Нека $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$ е произволен ДКА. Тогава следните условия са еквивалентни:

(1) $\mathcal{L}(\mathcal{A})$ е *безкраен*;

(2) $\mathcal{A}$ разпознава дума α, за която $|Q| \leq |\alpha| < 2|Q|$.

Доказателство. Да положим $L = \mathcal{L}(\mathcal{A})$. Да разгледаме двете посоки на твърдението.

$(1) \Rightarrow (2)$**.** 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 лемата за покачването, 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 $xz \in L$.

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

□

Сега вече много лесно можем да видим, че следните проблеми са алгоритмично разрешими за автоматните езици.

(Проблемът за празнота) Според *Твърдение 2.13*, *Алгоритъм (2)* разрешава проблемът за празнота за автоматни езици.

Algorithm 2 Проблемът за празнота за автоматни езици

```
1: procedure IsEmpty(A)
2:     for all n < |Q| do
3:         for all ω ∈ Σ^n do
4:             if belong(A, ω) then
5:                 return True
6:     return False
```

(Проблемът за включване) Проблемът за включването се свежда към проблема за празнота, защото за всеки два езика L_1 и L_2 е изпълнена веригата от еквивалентности:

$$\begin{aligned} L_1 \subseteq L_2 &\Leftrightarrow (L_1 \setminus L_2) = \emptyset \\ &\Leftrightarrow L_1 \cap \overline{L}_2 = \emptyset. \end{aligned}$$

Algorithm 3 Проблемът за включване за автоматни езици

```
1: procedure isIncluded(A, B)
2:     B_1 := complement(B)
3:     A_1 := intersect(A, B_1)
4:     return isEmpty(A_1)
```

(Проблемът за равенство) Понеже е изпълнена еквивалентността

$$L_1 = L_2 \Leftrightarrow L_1 \subseteq L_2 \,\&\, L_2 \subseteq L_1,$$

то е ясно, че проблемът за равенство на два автоматни езика е алгоритмично разрешим, защото просто трябва да приложим два пъти *Алгоритъм (3)*.

(Проблемът за безкрайност) От *Твърдение 2.14* директно получаваме следния прост алгоритъм, който разрешава проблемът за безкрайност на автоматен език.

Algorithm 4 Проблемът за безкрайност за автоматни езици

```
1: procedure isInfinite(A)
2:     for all n := |Q|, ..., 2|Q| − 1 do
3:         for all ω ∈ Σ^n do
4:             if belong(A, ω) then
5:                 return True
6:     return False
```

(Проблемът за универсалност) Този проблем отново е алгоритмично разрешим, защото имаме еквивалентността:

$$L = \Sigma^\star \Leftrightarrow \overline{L} = \emptyset.$$

Примерни задачи

Задача 2.18. 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 (∀) нямаме власт над това как избираме съответния елемент.
На стъпка от вида (∃) имаме тази власт. Тогава трябва да посочим конкретен елемент.

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

(∃) 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 $|\alpha| \geq p$.

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

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

(∃) 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

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

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

Доказателство. Доказателството следва стъпките:

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

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

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

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

(∃) 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 $|\omega > p + 1$, защото искаме да гарантираме, че $|xz| > 1$.

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

Задача 2.20. 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 $\omega = a^{p^2}$.

($\forall$) Разглеждаме произволно разбиване на ω на три части, $\omega = xyz$, 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 2.21. 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. В този случай, добра работа ще ни свърши думата $\omega = a^{(p+1)!}$.

($\forall$) 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

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

Възможно е да вземем $\omega = a^{(p+2)!}$. 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 $|y| \leq p$.

Да разгледаме $i = 2$. Тогава:

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

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

Задача 2.22. 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

$$L_2 = L_1 \cap \mathcal{L}[\![\boldsymbol{a^\star ba^\star b}]\!] = \{a^n b a^n b \mid n \in \mathbb{N}\}.$$

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. □

Задача 2.23. Докажете, че езикът $L = \{a^n b^k \mid n \neq k\}$ не е регулярен.

Упътване. Доказателството следва стъпките:

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

(∃) Избираме достатъчно дълга дума, която принадлежи на езика L. В този случай, добра работа ще ни свърши думата $\omega = a^p \cdot b^{p+p!}$.

(∀) Разглеждаме произволно разбиване на ω на три части, $\omega = xyz$, като $|xy| \leq p$ и $|y| \geq 1$. Да обърнем внимание, че $1 \leq |y| \leq p$. Ясно е, че $y = a^\ell$, за някое ℓ.

Тук е важно да отбележим, че изборът на i зависи от избора на разбиването xyz.

(∃) Да изберем числото $i = \frac{p!}{\ell}$. Тогава е ясно, че $y^i = a^{p!}$. Заключаваме, че

$$xy^{i+1}z = a^{p+p!} \cdot b^{p+p!} \notin L.$$

□

Упътване. Можем и далеч по-лесно да решим тази задача. Знаем, че регулярните езици са затворени относно допълнение. Следователно, ако допуснем, че L е регулярен, то езикът

$$L' = \{a\}^\star \cdot \{b\}^\star \setminus L$$

също трябва да е регулярен. Но $L' = \{a^n b^n \mid n \in \mathbb{N}\}$, за който вече знаем, че не е регулярен. □

Пример, за който критерият не е приложим

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 2.10. 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.

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 $xy^i z \in L$.

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

$$L_1 = L \cap \mathcal{L}[\![\boldsymbol{ca^\star b^\star}]\!] = \{ca^n b^n \mid n \in \mathbb{N}\}$$

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 $p = 1$.

(∀) Сега трябва да разгледаме всички думи $\alpha \in L$, за които $|\alpha| \geq 1$.

(∃) 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 $xy^i z \in L$.

□

2.10 Изоморфни автомати

Да разгледаме два произволни ДКА

$$\mathcal{A}_1 = (\Sigma, Q_1, q'_{\texttt{start}}, \delta_1, F_1),$$
$$\mathcal{A}_2 = (\Sigma, Q_2, q''_{\texttt{start}}, \delta_2, F_2).$$

[13, стр. 89]. Естествено, ние можем да дефинираме и изоморфни НКА, но за нашите цели е достатъчно да разгледаме само изоморфни ДКА.

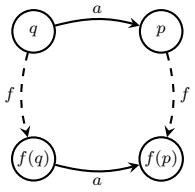
Казваме, че $\mathcal{A}_1$ и $\mathcal{A}_2$ са **изоморфни**, което означаваме с $\mathcal{A}_1 \cong \mathcal{A}_2$, ако съществува *биективна* функция $f : Q_1 \to Q_2$, за която:

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

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

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

Условие (3) може да се представи графично така:



С други думи, два автомата са изоморфни точно тогава, когато те са идентични с точност до преименуване на състоянията.

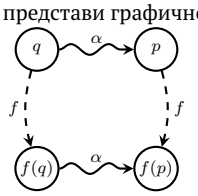
Ще казваме, че f задава изоморфизъм на $\mathcal{A}_1$ върху $\mathcal{A}_2$ и ще означаваме $\mathcal{A}_1 \cong_f \mathcal{A}_2$ или $f : \mathcal{A}_1 \cong \mathcal{A}_2$.

Твърдение 2.15. Нека $\mathcal{A}_1 \cong_f \mathcal{A}_2$. Тогава за всяка дума α и състояние $q \in Q_1$ е изпълнена еквивалентността:

$$f(\delta_1^\star(q, \alpha)) = \delta_2^\star(f(q), \alpha). \tag{2.10}$$

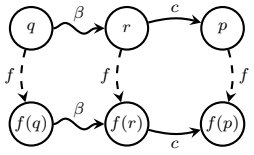
Доказателство. Както винаги, ще докажем *Свойство 2.10* с индукция по дължината на думата α.

Свойство 2.10 може да се представи графично така:



- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Ясно е, че *Свойство 2.10* 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 2.10* 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 имаме следната верига от равенства:

Индукционната стъпка може да се представи така:



$$\begin{aligned}
f(\delta_1^\star(q, \beta c)) &= f(\delta_1(\overbrace{\delta_1^\star(q, \beta)}^{p}, c)) \\
&= f(\delta_1(p, c)) && \text{// } \textit{Свойство (3)} \\
&= \delta_2(f(p), c) \\
&= \delta_2(f(\delta_1^\star(q, \beta), c)) \\
&= \delta_2(\delta_2^\star(f(q), \beta), c) && \text{// } \textbf{(И.П.)} \text{ за } \beta \\
&= \delta_2^\star(f(q), \beta c).
\end{aligned}$$

□

Твърдение 2.16. Ако $\mathcal{A}_1 \cong \mathcal{A}_2$, то $\mathcal{L}(\mathcal{A}_1) = \mathcal{L}(\mathcal{A}_2)$.

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

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

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}_1) &\Leftrightarrow \delta_1^\star(q'_{\texttt{start}}, \alpha) \in F_1 && \text{// деф. на } \mathcal{L}(\mathcal{A}_1)\\
&\Leftrightarrow f(\delta_1^\star(q'_{\texttt{start}}, \alpha)) \in F_2 && \text{// } \textit{Свойство (2)}\\
&\Leftrightarrow \delta_2^\star(f(q'_{\texttt{start}}), \alpha)) \in F_2 && \text{// } \textit{Твърдение 2.15}\\
&\Leftrightarrow \delta_2^\star(q''_{\texttt{start}}, \alpha) \in F_2 && \text{// } f(q'_{\texttt{start}}) \overset{\text{деф}}{=} q''_{\texttt{start}}\\
&\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}_2). && \text{// деф. на } \mathcal{L}(\mathcal{A}_2)
\end{aligned}$$

□

2.11 Минимален автомат

Сега ще видим, че автоматът на Бжозовски $\mathcal{B}_L$ за даден регулярен език L е в известен смисъл най-добрият възможен. Накратко, $\mathcal{B}_L$ 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.

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 $q_\alpha \overset{\text{деф}}{=} \delta^\star(q_{\texttt{start}}, \alpha)$. 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 2.17. Нека $L = \mathcal{L}(\mathcal{A})$. Тогава за всяка дума α е изпълнено:

$$\mathcal{L}_\mathcal{A}(q_\alpha) = \alpha^{-1}(L).$$

Доказателство. За произволна дума ω имаме следната верига от еквивалентности:

$$\begin{aligned}
\omega \in \mathcal{L}_\mathcal{A}(q_\alpha) &\Leftrightarrow \delta^\star(q_\alpha, \omega) \in F && \text{// от деф. на } \mathcal{L}_\mathcal{A}(q_\alpha)\\
&\Leftrightarrow \delta^\star(\delta^\star(q_{\texttt{start}}, \alpha), \omega) \in F && \text{// } q_\alpha \overset{\text{деф}}{=} \delta^\star(q_{\texttt{start}}, \alpha)\\
&\Leftrightarrow \delta^\star(q_{\texttt{start}}, \alpha\omega) \in F && \text{// } \textit{Твърдение 2.1}\\
&\Leftrightarrow \alpha\omega \in \mathcal{L}(\mathcal{A}) && \text{// деф. на } \mathcal{L}(\mathcal{A})\\
&\Leftrightarrow \alpha\omega \in L && \text{// } L = \mathcal{L}(\mathcal{A})\\
&\Leftrightarrow \omega \in \alpha^{-1}(L). && \text{// деф. на } \alpha^{-1}(L)
\end{aligned}$$

□

Твърдение 2.18. Нека L е произволен език. Тогава за произволно състояние M на $\mathcal{B}_L$ имаме равенството:

$$\mathcal{L}_{\mathcal{B}_L}(\underbrace{M}_{\text{състояние}}) = \underbrace{M}_{\text{език}}.$$

Доказателство. Понеже за произволна дума α имаме еквивалентностите

$$\begin{aligned}
\alpha \in \mathcal{L}_{\mathcal{B}_L}(M) &\Leftrightarrow \delta^\star_L(M, \alpha) \in F_L\\
&\Leftrightarrow \varepsilon \in \alpha^{-1}(M) && \text{// деф. на } F_L\\
&\Leftrightarrow \alpha \in M, && \text{// от } \textit{Задача 2.8}
\end{aligned}$$

заключаваме, че $\mathcal{L}_{\mathcal{B}_L}(M) = M$. □

Твърдение 2.19. 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 2.6. Нека L е регулярен език и $\mathcal{A}$ е произволен ДКА, за който $L = \mathcal{L}(\mathcal{A})$. Тогава $|Q_L| \leq |Q_\mathcal{A}|$.

Тази лема ни казва, че автоматът на Бжозовски има възможно най-малкия брой състояния измежду всички ДКА разпознаващи L [23, стр. 113].

Доказателство. Да разгледаме функцията $f : Q_\mathcal{A} \to Q_L$ зададена по следния начин:

$$f(q) \stackrel{\text{деф}}{=} \mathcal{L}_\mathcal{A}(q). \tag{2.11}$$

Първо, да видим защо f е добре дефинирана функция, т.е. да видим защо за всяко състояние $q \in Q_\mathcal{A}$, имаме $f(q) \in Q_L$. Да напомним, че приехме още в началото на раздела, че $\mathcal{A}$ е свързан автомат. Това означава, че за всяко състояние p, съществува дума α, за която $p = \delta^\star_\mathcal{A}(q^\mathcal{A}_{\texttt{start}}, \alpha)$, т.е. $p = q_\alpha$ според означението, което въведохме в началото на *Раздел 2.11*. Тогава от *Твърдение 2.17* следва, че $f(q_\alpha) = \alpha^{-1}(L) \in Q_L$, защото $\alpha^{-1}(L) = \mathcal{L}_\mathcal{A}(q_\alpha)$.

Второ, да видим, че f е сюрективна функция. За тази цел, да разгледаме произволно състояние $M \in Q_L$, което означава, че има дума α, за която $M = \alpha^{-1}(L)$. Отново според *Твърдение 2.17*,

$$f(q_\alpha) = \mathcal{L}_\mathcal{A}(q_\alpha) = \alpha^{-1}(M).$$

Сега от *Твърдение 2.19* можем да заключим, че

$$|Q_L| \leq |Q_\mathcal{A}|.$$

□

Следствие 2.5 (Критерий за регулярност на език)**.** Един език L е регулярен точно тогава, когато $\mathcal{B}_L$ има крайно много състояния.

Доказателство. Ако сме избрали алгебричния подход описан в *Раздел 2.8*, то ние отдавна знаем този критерий формулиран като *Следствие 2.5*. □

Доказателство. Ако сме избрали автоматния подход описан в *Раздел 2.4*, то ние чак сега можем да докажем този критерий. Нека L е регулярен. Тогава от *Теорема 2.2*, $L = \mathcal{L}(\mathcal{A})$ за някой ДКА $\mathcal{A}$, защото НКА разпознават същите езици като ДКА. Да напомним, че от *Твърдение 2.12* също имаме и $\mathcal{L}(\mathcal{B}_L) = L$. Понеже от *Лема 2.6* имаме, че $|Q_L| \leq |Q_\mathcal{A}|$, то $\mathcal{B}_L$ има краен брой състояния.

Обратно, ако $\mathcal{B}_L$ има крайно много състояния, то тогава $\mathcal{B}_L$ е ДКА. Понеже $\mathcal{L}(\mathcal{B}_L) = L$ според *Твърдение 2.12*, то L е автоматен и следователно регулярен по теоремата на Клини, за която имаме две доказателства - автоматен подход и алгебричен подход. □

Следствие 2.6. Нека L е регулярен език. Тогава автоматът $\mathcal{B}_L$, построен по метода на Бжозовски за L, има *минималния* възможен брой състояния измежду всички детерминирани крайни автомати разпознаващи L.

Следващото твърдение е до голяма степен очевидно, но за пълнота на изложението, ще го разгледаме подробно.

Твърдение 2.20. Нека A и B са крайни равномощни множества. Докажете, че ако $g : A \to B$ е сюрекция, то g е биекция.

Доказателство. Нека $B = \{b_0, \ldots, b_{n-1}\}$. За всеки индекс $i < n$ да положим

$$A_i \stackrel{\text{деф}}{=} \{a \in A \mid g(a) = b_i\}.$$

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

$$n = |A| = |\bigcup_{i<n} A_i| = \sum_{i<n} |A_i|.$$

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 $|A_i| > 1$. □

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. Това твърдение трябва да може да докажете сами! Да напомним, че от курса по Дискретна математика имаме формулата $|X \cup Y| = |X| + |Y| + |X \cap Y|$. Просто в нашия случай $|X \cap Y| = 0$.

Теорема 2.7. За всеки регулярен език L съществува *единствен* минимален ДКА с точност до изоморфизъм.

С други думи, ако имаме два минимални ДКА за L, то можем да получим единия автомат от другия чрез внимателно преименуване на състоянията.

Доказателство. Вече знаем, че автоматът $\mathcal{B}_L$, построен по метода на Бжозовски за L, има минималния възможен брой състояния. Нека $\mathcal{A}$ е друг ДКА разпознаващ L и $|Q_\mathcal{A}| = |Q_L|$. Трябва да докажем, че $\mathcal{A} \cong \mathcal{B}_L$. 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 $f : Q_\mathcal{A} \to Q_L$, където

$$f(p) \overset{\text{деф}}{=} \mathcal{L}_\mathcal{A}(p). \tag{2.12}$$

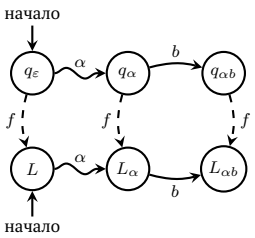
От *Лема 2.6* знаем, че f е сюрективна. Понеже $|Q_\mathcal{A}| = |Q_L|$, то от *Твърдение 2.20* имаме, че f е всъщност биекция. Остава да видим защо $\mathcal{A} \cong_f \mathcal{B}$. Да напомним, че можем да разглеждаме състоянията на $Q_\mathcal{A}$ като q_α и тогава $\mathcal{L}_\mathcal{A}(q_\alpha) = \alpha^{-1}(L)$ от *Твърдение 2.17*.

- За началното състояние имаме, че:

$$\begin{aligned} f(q^\mathcal{A}_{\texttt{start}}) &= f(q_\varepsilon) && \text{// } q^\mathcal{A}_{\texttt{start}} = q_\varepsilon \\ &= \mathcal{L}_\mathcal{A}(q_\varepsilon) && \text{// от деф. на } f \\ &= \varepsilon^{-1}(L) && \text{// } \textit{Твърдение 2.17} \\ &= L. \end{aligned}$$

- За финалните състояния имаме, че:

$$\begin{aligned} q_\alpha \in F_\mathcal{A} &\Leftrightarrow \delta^\star_\mathcal{A}(q_{\texttt{start}}, \alpha) \in F_\mathcal{A} && \text{// } q_\alpha = \delta^\star_\mathcal{A}(q_{\texttt{start}}, \alpha) \\ &\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}) && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \varepsilon \in \alpha^{-1}(L) && \text{// } L = \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \alpha^{-1}(L) \in F_L. && \text{// деф. на } F_L \end{aligned}$$



Тук сме означили

$L_\alpha = \alpha^{-1}(L)$
$L_{\alpha b} = (\alpha b)^{-1}(L)$.

- Остава да докажем, че за произволна буква b и произволни състояния $q, p \in Q_\mathcal{A}$ е изпълнено, че:

$$\delta_L(f(q), b) = f(\delta_\mathcal{A}(q, b)),$$

или с други думи,

$$\delta_L(\mathcal{L}_\mathcal{A}(q), b) = \mathcal{L}_\mathcal{A}(\delta_\mathcal{A}(q, b)). \tag{2.13}$$

Това се вижда директно от веригата от еквивалентности:

$$\begin{aligned} \omega \in \delta_L(\mathcal{L}_\mathcal{A}(q), b) &\Leftrightarrow \omega \in b^{-1}(\mathcal{L}_\mathcal{A}(q)) \\ &\Leftrightarrow b\omega \in \mathcal{L}_\mathcal{A}(q) \\ &\Leftrightarrow \delta^\star_\mathcal{A}(q, b\omega) \in F_\mathcal{A} \\ &\Leftrightarrow \delta^\star_\mathcal{A}(\delta_\mathcal{A}(q, b), \omega) \in F_\mathcal{A} \\ &\Leftrightarrow \omega \in \mathcal{L}_\mathcal{A}(\delta^\star_\mathcal{A}(q, b)). \end{aligned}$$

Така доказахме, че f задава изоморфизъм между $\mathcal{A}$ и $\mathcal{B}_L$. □

Следствие 2.7 (Критерий за минималност)**.** Нека $\mathcal{A}$ е *свързан* ДКА разпознаващ регулярния език L. Тогава $\mathcal{A}$ е *минимален* автомат за езика L точно тогава, когато е изпълнена импликацията:

$$(\forall p \in Q)(\forall q \in Q)[\, p \neq q \implies \mathcal{L}_{\mathcal{A}}(p) \neq \mathcal{L}_{\mathcal{A}}(q)\,]. \tag{2.14}$$

Доказателство. Нека $\mathcal{B}_L$ е автоматът на Бжозовски за L и да разгледаме функцията $f : Q_{\mathcal{A}} \to Q_L$ дефинирана като $f(q) = \mathcal{L}_{\mathcal{A}}(q)$.

Първо, нека $\mathcal{A}$ е минимален автомат за L. Тогава знаем от *Теорема 2.7*, че f е биекция. От инективността на f следва, че имаме импликцията (2.14).

Второ, нека импликацията (2.14) е изпълнена. Това означава, че f е инективна функция, откъдето имаме, че $|Q_{\mathcal{A}}| \leq |Q_L|$. Понеже $\mathcal{B}_L$ е минимален автомат, то $\mathcal{A}$ също е минимален автомат. □

2.12 Критерий за регулярност (алгебричен подход)

За да докажем, че един език L не е регулярен можем да приложим *Следствие 2.5* като докажем, че автоматът на Бжозовски за L има безкрайно много състояния. Обърнете внимание, че не е нужно да намерим всички състояния на автомата $\mathcal{B}_L$, а само това, че са безкрайно много.

Сравнете с *Пример 2.8*.

Пример 2.11. 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:

$$k < m \implies (a^k)^{-1}(L) \neq (a^m)^{-1}(L).$$

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 2.12. 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 2.13. 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 2.24. Докажете, че езикът

$$L = \{a^{f_n} \mid f_0 = f_1 = 1 \ \& \ f_{n+2} = f_{n+1} + f_n\}$$

не е регулярен.

2.13 Автомат на Майхил-Нероуд

В почти всички учебници се разглежда конструкцията на Майхил-Нероуд вместо тази на Бжозовски. Тук накратко ще покажем, че двете конструкции са идентични, т.е. автоматите получени по двата метода са изоморфни.

на англ. Myhill-Nerode

Определение 2.5. Нека L е произволен 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,

$$\alpha \approx_L \beta \Leftrightarrow (\forall \omega \in \Sigma^\star)[\, \alpha\omega \in L \Leftrightarrow \beta\omega \in L \,].$$

Това означава, че ако искаме да докажем, че $\alpha \not\approx_L \beta$, то е достатъчно да намерим дума ω, за която $\alpha\omega \in L$, но $\beta\omega \notin L$, или обратно, $\beta\omega \in L$, но $\alpha\omega \notin L$.

$\approx_L$ е известна като релация на Майхил-Нероуд или просто като релация на Нероуд.

Лесно се съобразява, че $\approx_L$ е релация на еквивалентост. За произволна дума α, да означим с $[\alpha]_L$ класът на еквивалентост на α, т.е.

$$[\alpha]_L \overset{\text{деф}}{=} \{\beta \in \Sigma^\star \mid \alpha \approx_L \beta\}.$$

Твърдение 2.21. Нека L е произволен език. За всеки две думи α, β и буква x е изпълнена импликацията:

$$\alpha \approx_L \beta \implies \alpha x \approx_L \beta x.$$

Доказателство. Нека $\alpha \approx_L \beta$, т.е. $\alpha^{-1}(L) = \beta^{-1}(L)$. Тогава $\alpha x \approx_L \beta x$, защото:

$$\begin{aligned}
(\alpha x)^{-1}(L) &= x^{-1}(\alpha^{-1}(L)) && \text{// } \textit{Твърдение 2.10} \\
&= x^{-1}(\beta^{-1}(L)) && \text{// } \alpha^{-1}(L) = \beta^{-1}(L) \\
&= (\beta x)^{-1}(L). && \text{// } \textit{Твърдение 2.10}
\end{aligned}$$

□

Твърдение 2.22. Нека L е произволен език. За всяка дума α е изпълнена еквивалентността:

$$\alpha \in L \Leftrightarrow [\alpha]_L \subseteq L.$$

Доказателство. За посоката ($\Rightarrow$), нека $\alpha \in L$ и да вземем произволна дума $\beta \in [\alpha]_L$. Ще видим, че $\beta \in L$. И така, щом $\beta \in [\alpha]_L$, то $\alpha^{-1}(L) = \beta^{-1}(L)$. Понеже $\alpha \in L$, то $\varepsilon \in \alpha^{-1}(L)$, което означава, че $\varepsilon \in \beta^{-1}(L)$. Оттук заключаваме, че $\beta \in L$.

За посоката ($\Leftarrow$), щом $[\alpha]_L \subseteq L$ и $\alpha \in [\alpha]_L$, то е ясно, че $\alpha \in L$. □

Ще наричаме $\mathcal{M}$ автомат на Майхил-Нероуд. На практика във всеки учебник се разглежда автомата на Майхил-Нероуд вместо автомата на Бжозовски. Например, [18, стр. 98], [10, стр. 65], [25, стр. 91], [13, стр. 89]

Определение 2.6. За даден език L, дефинираме **автоматът на Майхил-Нероуд** $\mathcal{M}_L = \langle \Sigma, Q_{\mathcal{M}_L}, q^{\mathcal{M}_L}_{\mathtt{start}}, \delta_{\mathcal{M}_L}, F_{\mathcal{M}_L} \rangle$ за езика L по следния начин:

- $Q_{\mathcal{M}_L} \stackrel{\text{деф}}{=} \{\, [\alpha]_L \mid \alpha \in \Sigma^\star \,\}$;
- $q^{\mathcal{M}_L}_{\mathtt{start}} \stackrel{\text{деф}}{=} [\varepsilon]_L$;
- $F_{\mathcal{M}_L} \stackrel{\text{деф}}{=} \{\, [\alpha]_L \mid [\alpha]_L \subseteq L \,\}$;
- $\delta_{\mathcal{M}_L}([\alpha]_L, b) \stackrel{\text{деф}}{=} [\alpha b]_L$, за всяка $\alpha \in \Sigma^\star$ и $b \in \Sigma$.

Трябва да проверим, че нашата дефиниция на $\delta_{\mathcal{M}}$ не зависи от избора на представител от класа $[\alpha]_L$, който сме направили. По-формално, нека $[\alpha]_L = [\beta]_L$, т.е. $\alpha \approx_L \beta$. Трябва да докажем, че $\delta_{\mathcal{M}_L}([\alpha]_L, b) = \delta_{\mathcal{M}_L}([\beta]_L, b)$, т.е. $[\alpha b]_L = [\beta b]_L$. Но това е лесно да се види, защото от *Твърдение 2.21* знаем, че $\approx_L$ е дясно-инвариантна релация на еквивалентност. Това означава, че $\alpha b \approx_L \beta b$ или с други думи, $[\alpha b]_L = [\beta b]_L$.

Лема 2.7. За всеки език L, $\mathcal{M}_L \cong \mathcal{B}_L$.

Доказателство. Да разгледаме функцията $f : Q_{\mathcal{M}_L} \to Q_L$, където:

$$f([\alpha]_L) \stackrel{\text{деф}}{=} \alpha^{-1}(L).$$

Ще докажем, че f задава биекция между $\mathcal{M}_L$ и $\mathcal{B}_L$.

- Първо, съобразете сами, че f е добре дефинирана.
- Да видим, че f е биективна. Нека $[\alpha]_L \neq [\beta]_L$, което означава, че $\alpha^{-1}(L) \neq \beta^{-1}(L)$. Оттук веднага следва, че $f([\alpha]_L) \neq f([\beta]_L)$. За сюрективността, нека $M \in Q_L$. Това означава, че $M = \alpha^{-1}(L)$, за някоя дума α. Тогава, по дефиниция, $f([\alpha]_L) = M$.
- Ясно е, че $f(q_{\mathtt{start}}^{\mathcal{M}_L}) = q_{\mathtt{start}}^{L}$.
- За да видим, че $[\alpha]_L \in F_{\mathcal{M}_L} \Leftrightarrow f([\alpha]_L) \in F_L$ е достатъчно да проследим еквивалентностите:

$$\begin{aligned}
[\alpha]_L \in F_{\mathcal{M}_L} &\Leftrightarrow [\alpha]_L \subseteq L \\
&\Leftrightarrow \alpha \in L \\
&\Leftrightarrow \varepsilon \in \alpha^{-1}(L) \\
&\Leftrightarrow \alpha^{-1}(L) \in F_L \\
&\Leftrightarrow f([\alpha]_L) \in F_L.
\end{aligned}$$

- Остава да видим, че $f(\delta_{\mathcal{M}_L}([\alpha]_L, b)) = \delta_L(f([\alpha]_L), b)$. Достатъчно е да проследим веригата от равенства:

$$\begin{aligned}
f(\delta_{\mathcal{M}_L}([\alpha]_L, b)) &= f([\alpha b]_L) \\
&= (\alpha b)^{-1}(L) \\
&= b^{-1}(\alpha^{-1}(L)) \\
&= \delta_L(\alpha^{-1}(L), b) \\
&= \delta_L(f([\alpha]_L), b).
\end{aligned}$$

□

Теоремата е доказана независимо от Майхил [15] и Нероуд [16].

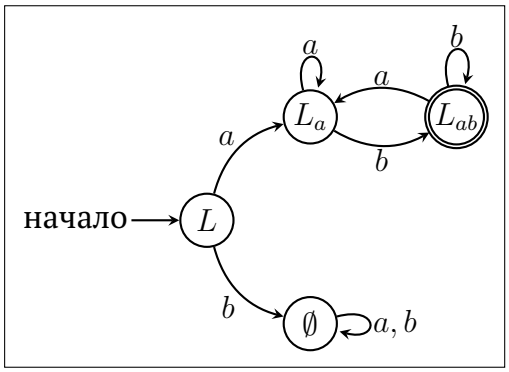
Теорема 2.8 (Майхил-Нероуд)**.** Един език L е регулярен точно тогава, когато $\mathcal{M}_L$ има краен брой състояния. Освен това, $\mathcal{M}_L$ е минимален ДКА за L.

Доказателство. Понеже от *Лема 2.7* имаме, че $\mathcal{B}_L \cong \mathcal{M}_L$, то твърдението следва директно от *Следствие 2.5*. □

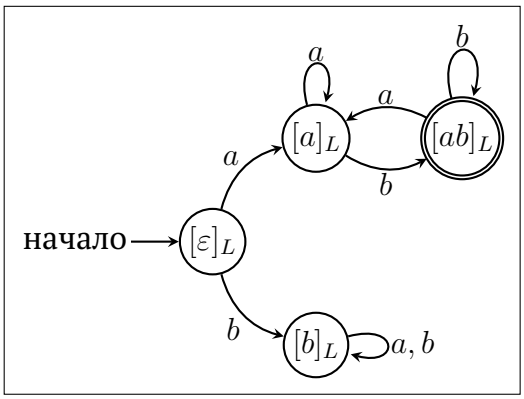
Пример 2.14. Да разгледаме езика $L = \mathcal{L}[\![\boldsymbol{a} \cdot (\boldsymbol{a} + \boldsymbol{b})^\star \cdot \boldsymbol{b}]\!]$.

Да напомним, че в *Пример 2.13* вече видяхме автоматът на Бжозовски за L. За да получим по-компактен запис, тук пишем

$$L_a \stackrel{\text{деф}}{=} a^{-1}(L)$$
$$L_{ab} \stackrel{\text{деф}}{=} (ab)^{-1}(L).$$



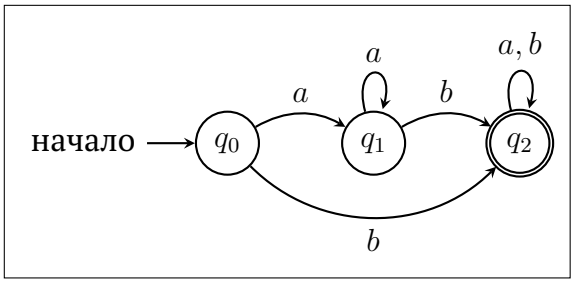
(а) Автомат на Бжозовски за L



(б) Автомат на Майхил-Нероуд за L

Основната идея зад конструкцията за автомат на Бжозовски е, че с всяко състояния на автомата асоциираме езика на състоянието. Следващия пример илюстрира как тази идея може да ни помогне в разсъжденията как можем да „компресираме" даден автомат, така че да получим еквивалентен на него минимален.

Пример 2.15. Да разгледаме автомата на *Фигура 2.29*.

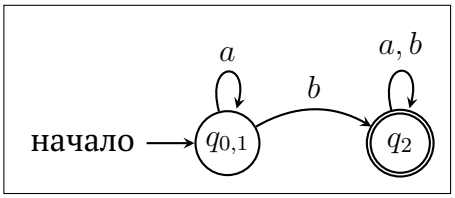


Фигура 2.29: $\mathcal{L}(\mathcal{A}) = \mathcal{L}[\![\boldsymbol{a^\star b(a+b)^\star}]\!]$.

От *Пример 2.7* знаем, че

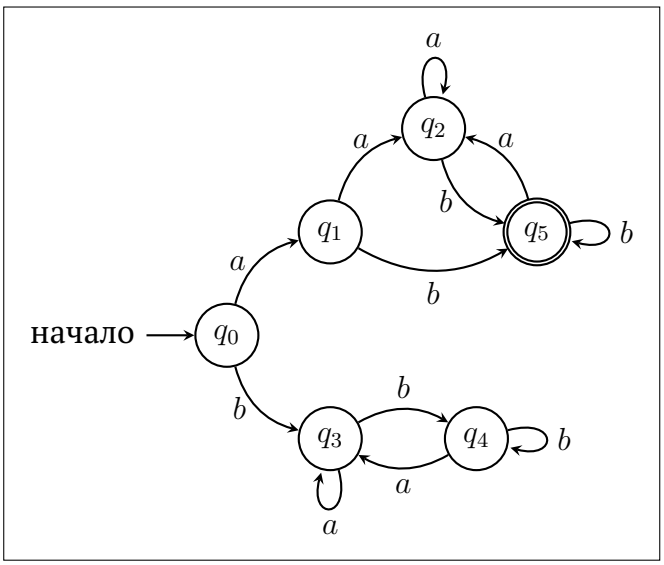
$$\mathcal{L}_{\mathcal{A}}(q_0) = \mathcal{L}_{\mathcal{A}}(q_1) = \mathcal{L}[\![\boldsymbol{a^\star \cdot b \cdot (a+b)^\star}]\!].$$

Това означава, че тези две състояния са неотличими, т.е. можем да ги „слеем" в едно състояние, което ще означим като $q_{0,1}$. Да видим какво получаваме:



Фигура 2.30: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \mathcal{L}[\![\boldsymbol{a^\star b(a+b)^\star}]\!]$.

Пример 2.16. 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.

2.14 Минимизация (автоматен подход)

Нека отново да приемем, че сме фиксирали един детерминиран краен автомат $\mathcal{A} = \langle \Sigma, Q, q_{\mathtt{start}}, \delta, F \rangle$ като всяко състояние е достижимо от началното. Да напомним, че 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 имаме, че

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(q_{\mathtt{start}}).$$

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

$$(\forall \omega \in \Sigma^\star)[\, \delta^\star(p, \omega) \in F \;\Leftrightarrow\; \delta^\star(q, \omega) \in F \,]. \tag{2.15}$$

Задача 2.25. Докажете, че релацията $\equiv_{\mathcal{A}}$ между състояния на автомата $\mathcal{A}$ е релация на еквивалентност.

Твърдение 2.23. Нека $\mathcal{A}$ е произволен ДКА. За всяко състояние q на $\mathcal{A}$ е изпълнено, че:

$$[q]_{\equiv_{\mathcal{A}}} \cap F \neq \emptyset \;\Leftrightarrow\; [q]_{\equiv_{\mathcal{A}}} \subseteq F.$$

Доказателство. Посоката ($\Leftarrow$) е очевидна. Да разгледаме посоката ($\Rightarrow$). Трябва да докажем, че всяко състояние $r \in [q]_{\equiv_{\mathcal{A}}}$ принадлежи на F. Да фиксираме едно състояние $p \in [q]_{\equiv_{\mathcal{A}}} \cap F$. Щом $p \in F$, то $\varepsilon \in \mathcal{L}_{\mathcal{A}}(p)$. Щом $p \in [q]_{\equiv_{\mathcal{A}}}$ и $r \in [q]_{\equiv_{\mathcal{A}}}$, то $p \equiv_{\mathcal{A}} r$ и следователно $\mathcal{L}_{\mathcal{A}}(p) = \mathcal{L}_{\mathcal{A}}(r)$. Вече знаем, че $\varepsilon \in \mathcal{L}_{\mathcal{A}}(p)$. Тогава е ясно, че $\varepsilon \in \mathcal{L}_{\mathcal{A}}(r)$. С други думи, $\delta^\star_{\mathcal{A}}(r, \varepsilon) \in F$, откъдето получаваме, че $r \in F$, защото, според дефиницията на $\delta^\star_{\mathcal{A}}$, имаме $\delta^\star_{\mathcal{A}}(r, \varepsilon) = r$. □

Твърдение 2.24. 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:

$$\begin{aligned} p \equiv_{\mathcal{A}} q &\Leftrightarrow \mathcal{L}_{\mathcal{A}}(p) = \mathcal{L}_{\mathcal{A}}(q) \\ &\Leftrightarrow (\forall \beta \in \Sigma^\star)[\delta^\star(p,\beta) \in F \Leftrightarrow \delta^\star(q,\beta) \in F] \\ &\Rightarrow (\forall \beta \in \Sigma^\star)[\delta^\star(p,a\beta) \in F \Leftrightarrow \delta^\star(q,a\beta) \in F] \\ &\Leftrightarrow (\forall \beta \in \Sigma^\star)[\delta^\star(\delta(p,a),\beta) \in F \Leftrightarrow \delta^\star(\delta(q,a),\beta) \in F] \\ &\Leftrightarrow \mathcal{L}_{\mathcal{A}}(\delta(p,a)) = \mathcal{L}_{\mathcal{A}}(\delta(q,a)) \\ &\Leftrightarrow \delta(p,a) \equiv_{\mathcal{A}} \delta(q,a). \end{aligned}$$

□

Определение 2.7. За детерминирания краен автомат $\mathcal{A} = \langle \Sigma, Q, q_{\mathtt{start}}, \delta, F \rangle$, дефинираме автомата

$$\mathcal{A}' = \langle \Sigma, Q', q'_{\mathtt{start}}, \delta', F' \rangle$$

по следния начин:

- $Q' \stackrel{\text{деф}}{=} \{[q]_{\equiv_{\mathcal{A}}} \mid q \in Q\}$;
- $q'_{\mathtt{start}} \stackrel{\text{деф}}{=} [q_{\mathtt{start}}]_{\equiv_{\mathcal{A}}}$;
- $\delta'([q]_{\equiv_{\mathcal{A}}}, a) \stackrel{\text{деф}}{=} [\delta(q,a)]_{\equiv_{\mathcal{A}}}$;
- $F' \stackrel{\text{деф}}{=} \{[q]_{\equiv_{\mathcal{A}}} \mid [q]_{\equiv_{\mathcal{A}}} \cap F \neq \emptyset\}$;

Задача 2.26. Докажете, че $\delta' : Q' \times \Sigma \to \Sigma$ е добре дефинирана функция.

Упътване. От *Твърдение 2.24* имаме, че

$$[p]_{\equiv_{\mathcal{A}}} = [q]_{\equiv_{\mathcal{A}}} \implies \delta'([p]_{\equiv_{\mathcal{A}}}, a) = \delta'([q]_{\equiv_{\mathcal{A}}}, a).$$

С други думи, дефиницията на δ' не зависи от избора на представител от класа $[q]_{\equiv_{\mathcal{A}}}$, който сме направили.

□

Твърдение 2.25. Нека $\mathcal{A}$ е произволен ДКА. За всяко състояние q на автомата $\mathcal{A}$ и дума α е изпълнено, че

$$\delta'^\star([q]_{\equiv_{\mathcal{A}}}, \alpha) = [\delta^\star(q,\alpha)]_{\equiv_{\mathcal{A}}}. \tag{2.16}$$

Доказателство. Ще докажем Свойство (2.16) 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 (2.16) е вярно за думи с дължина n.

- Нека $|\alpha| = n + 1$, т.е. $\alpha = \beta a$ и $|\beta| = n$. Тогава

$$\begin{aligned}
\delta'^{\star}([q]_{\equiv_{\mathcal{A}}}, \beta a) &= \delta'(\delta'^{\star}([q]_{\equiv_{\mathcal{A}}}, \beta), a) && \text{// деф. на } \delta'^{\star} \\
&= \delta'([\underbrace{\delta^{\star}(q, \beta)}_{p}]_{\equiv_{\mathcal{A}}}, a) && \text{// } \textbf{(И.П.)} \text{ за } \beta \\
&= \delta'([p]_{\equiv_{\mathcal{A}}}, a) \\
&= [\delta(p, a)]_{\equiv_{\mathcal{A}}} && \text{// деф. на } \delta' \\
&= [\delta(\delta^{\star}(q, \beta), a)]_{\equiv_{\mathcal{A}}} && \text{// } p = \delta'^{\star}(q, \beta) \\
&= [\delta^{\star}(q, \beta a)]_{\equiv_{\mathcal{A}}}. && \text{// деф. на } \delta^{\star}
\end{aligned}$$

□

Сега ще видим, че новата конструкция на автомата $\mathcal{A}'$ запазва езика на първоначалния автомат $\mathcal{A}$. Автоматът на Бжозовски (*Твърдение 2.12*) и автоматът на Майхил-Нероуд (*Определение 2.6*) са други конструкции с това свойство. Разликата е, че конструкцията на $\mathcal{A}'$ работи върху автомата $\mathcal{A}$ вместо върху езика на $\mathcal{A}$.

Лема 2.8. За всеки ДКА $\mathcal{A}$ имаме, че $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$.

Доказателство. Достатъчно е да проследим следните еквивалентности за произволна дума α:

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}) &\Leftrightarrow \delta^{\star}(q_{\mathtt{start}}, \alpha) \in F && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\
&\Leftrightarrow [\delta^{\star}(q_{\mathtt{start}}, \alpha)]_{\equiv_{\mathcal{A}}} \in F' && \text{// деф. на } F' \\
&\Leftrightarrow \delta'^{\star}([q_{\mathtt{start}}]_{\equiv_{\mathcal{A}}}, \alpha) \in F' && \text{// } \textit{Твърдение 2.25} \\
&\Leftrightarrow \delta'^{\star}(q'_{\mathtt{start}}, \alpha) \in F' && \text{// } q'_{\mathtt{start}} \stackrel{\text{деф}}{=} [q_{\mathtt{start}}]_{\equiv_{\mathcal{A}}} \\
&\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}'). && \text{// деф. на } \mathcal{L}(\mathcal{A}')
\end{aligned}$$

□

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 2.26. 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:

$$a_0 \equiv a_1 \overset{\text{деф}}{\Leftrightarrow} f(a_0) = f(a_1).$$

Дефинираме $[a]_\equiv$ да бъде класът на еквивалентност на a, т.е.

$$[a]_\equiv \overset{\text{деф}}{=} \{a_0 \in A \mid f(a) = f(a_0)\}.$$

Нека $A' \overset{\text{деф}}{=} \{[a]_\equiv \mid a \in A\}$. Тогава $f' : A' \to B$, където $f'([a]_\equiv) \overset{\text{деф}}{=} f(a)$ е биекция.

Доказателство. Достатъчно е да направим следните проверки:

Това твърдение го формулираме в общия случай, но ние ще го използваме, когато $\equiv$ е релацията $\equiv_\mathcal{A}$. Сравнете с *Твърдение 2.20*.

- 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 2.9. Автоматът $\mathcal{A}'$ е минимален ДКА за езика $\mathcal{L}(\mathcal{A})$.

Доказателство. Нека $\mathcal{B}_L$ е автоматът на Бжозовски за езика $L = \mathcal{L}(\mathcal{A})$. За да докажем, че $\mathcal{A}'$ е минимален ДКА за $\mathcal{L}(\mathcal{A})$, според *Лема 2.8* и *Теорема 2.7*, достатъчно е да докажем, че $|Q'| = |Q_L|$. От доказателството на *Лема 2.6* знаем, че функцията $f : Q_\mathcal{A} \to Q_L$ е сюрекция, където

Възможно е в доказателството да подходим и по друг начин. Ако докажем, че $\mathcal{A}' \cong_{f'} \mathcal{B}_L$, то оттук директно следва, че $\mathcal{A}'$ е минимален автомат разпознаващ L.

$$f(q) \overset{\text{деф}}{=} \mathcal{L}_\mathcal{A}(q).$$

Нека дефинираме $f' : Q' \to Q_L$ като

$$f'([q]_{\equiv_\mathcal{A}}) \overset{\text{деф}}{=} f(q).$$

От *Твърдение 2.26* имаме, че f' е биекция. Заключаваме, че $|Q'| = |Q_L|$. □

2.14.1 Алгоритъм за минимизация

При даден език L и детерминиран краен автомат $\mathcal{A} = \langle \Sigma, Q, q_{\mathtt{start}}, \delta, F \rangle$, 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 2.15), 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

$$\mathcal{L}^n_{\mathcal{A}}(p) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid |\omega| \leq n \;\&\; \delta^\star(p, \omega) \in F\}.$$

Можем ли да дадем горна граница m, така че $L(\mathcal{A}) = \bigcup_{n \leq m} \mathcal{L}^n_{\mathcal{A}}(q_{\mathtt{start}})$?

Лесно се съобразява, че

$$L(\mathcal{A}) = \bigcup_{n \geq 0} \mathcal{L}^n_{\mathcal{A}}(q_{\mathtt{start}}).$$

За всяко естествено число n, дефинираме бинарните релации $\equiv^n_{\mathcal{A}}$ върху Q по следния начин:

$$p \equiv^n_{\mathcal{A}} q \stackrel{\text{деф}}{\Leftrightarrow} \mathcal{L}^n_{\mathcal{A}}(p) = \mathcal{L}^n_{\mathcal{A}}(q).$$

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 2.27. За всеки две състояния p и q, и всяко естествено число n, $p \equiv^{n+1}_{\mathcal{A}} q$ точно тогава, когато:

а) $p \equiv^n_{\mathcal{A}} q$ и

б) $(\forall a \in \Sigma)[\delta(q, a) \equiv^n_{\mathcal{A}} \delta(p, a)]$.

Доказателство. Да положим $\Sigma^{\leq n} \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid |\alpha| \leq n\}$. Получаваме еквивалентностите: [18, 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 2.28. Нека $\equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{n+1}$. Тогава:

$$m > n \implies \equiv_{\mathcal{A}}^{n} = \equiv_{\mathcal{A}}^{m}. \tag{2.17}$$

Доказателство. Ще докажем Свойство (2.17) с индукция по m за $m > n$.

- Базата на индукцията е случаят $m = n+1$, за който Свойство (2.17) 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:

$$\begin{aligned}
p \equiv_{\mathcal{A}}^{m+1} q &\Leftrightarrow p \equiv_{\mathcal{A}}^{m} q \,\&\, (\forall a \in \Sigma)[\delta(p,a) \equiv_{\mathcal{A}}^{m} \delta(q,a)] && \text{// от } \textit{Твърдение 2.27} \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n} q \,\&\, (\forall a \in \Sigma)[\delta(p,a) \equiv_{\mathcal{A}}^{n} \delta(q,a)] && \text{// от } \textbf{(И.П.)} \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n+1} q && \text{// от } \textit{Твърдение 2.27} \\
&\Leftrightarrow p \equiv_{\mathcal{A}}^{n} q. && \text{// от условието}
\end{aligned}$$

□

Твърдение 2.29. 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 изпълнено, 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 2.28* 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 2.30. 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 2.28* и *Твърдение 2.29* 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 2.29* следва, че $\equiv_{\mathcal{A}}^{|Q|-1} = \equiv_{\mathcal{A}}$.

□

[24, стр. 83]

Algorithm 5 Кубичен алгоритъм за минимизация на ДКА

```
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 2.27
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
```

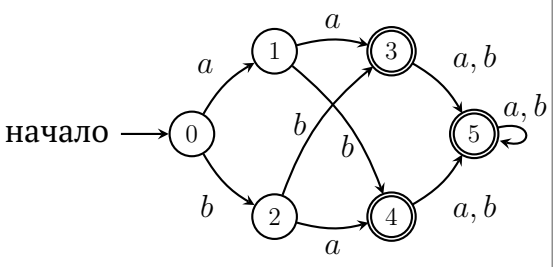
Задача 2.27. Докажете, че за всеки две състояния p и q,

$$p \equiv_{\mathcal{A}} q \;\Leftrightarrow\; \texttt{E[p][q] == true}.$$

Примерни задачи

[13, стр. 79]

Задача 2.28. Постройте минимален автомат $\mathcal{A}'$ разпознаващ езика на детерминирания краен автомат $\mathcal{A}$.



Фигура 2.31: 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 2.29*, 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 $\equiv^0_{\mathcal{A}}$.

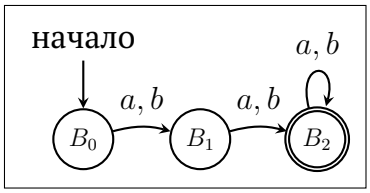
$\equiv^0_{\mathcal{A}}$	A_0			A_1		
Q	0	1	2	$3^\star$	$4^\star$	$5^\star$
a	A_0	A_1	A_1	A_1	A_1	A_1
b	A_0	A_1	A_1	A_1	A_1	A_1

Като използваме *Твърдение 2.27*, 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 2.32: Минимален 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

$$\begin{aligned}\mathcal{L}_{\mathcal{A}}(3) &= \{\varepsilon\} \cup \{a,b\} \cdot \mathcal{L}_{\mathcal{A}}(5)\\ &= \{\varepsilon\} \cup \{a,b\} \cdot \{a,b\}^\star\\ &= \{a,b\}^\star\\ \mathcal{L}_{\mathcal{A}}(4) &= \{\varepsilon\} \cup \{a,b\} \cdot \mathcal{L}_{\mathcal{A}}(5)\\ &= \{a,b\}^\star.\end{aligned}$$

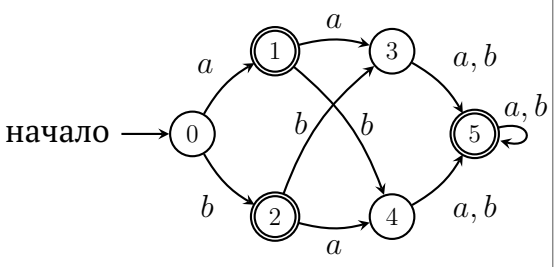
Така получаваме, че $\mathcal{L}_{\mathcal{A}}(3) = \mathcal{L}_{\mathcal{A}}(4) = \mathcal{L}_{\mathcal{A}}(5)$.

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

$$\mathcal{L}_{\mathcal{A}}(1) = \{a\} \cdot \mathcal{L}_{\mathcal{A}}(3) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(4)$$
$$\mathcal{L}_{\mathcal{A}}(2) = \{a\} \cdot \mathcal{L}_{\mathcal{A}}(4) \cup \{b\} \cdot \mathcal{L}_{\mathcal{A}}(3).$$

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 2.29. Постройте минимален автомат $\mathcal{A}'$ разпознаващ $\mathcal{L}(\mathcal{A})$.



Фигура 2.33: 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 2.27*.

$\equiv^0_{\mathcal{A}}$	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:

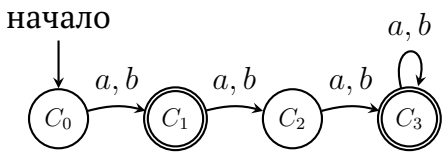
$$C_0 \stackrel{\text{деф}}{=} \{0\},$$
$$C_1 \stackrel{\text{деф}}{=} \{1,2\},$$
$$C_2 \stackrel{\text{деф}}{=} \{3,4\},$$
$$C_3 \stackrel{\text{деф}}{=} \{5\}.$$

- 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 2.34* за преходите на минималния автомат, които могат да се представят и таблично 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 2.34: 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 $\{\omega \in \{a,b\}^\star \mid |\omega| \neq 0,2\}$.

2.15 Минимизация (алгебричен подход)

[24, стр. 88]

Нека $\mathcal{A} = (\Sigma, Q, \delta, q_{\texttt{start}}, F)$ е детерминиран краен автомат. Тогава полагаме

$$\mathcal{A}_q \stackrel{\text{деф}}{=} (\Sigma, Q, \delta, q, F).$$

Нека $\mathcal{A} = (\Sigma, Q, \Delta, Q_{\texttt{start}}, F)$ е недетерминиран краен автомат. Тогава полагаме

$$\mathcal{A}_q \stackrel{\text{деф}}{=} (\Sigma, Q, \Delta, \{q\}, F).$$

За произволен краен автомат $\mathcal{A}$, за да не пишем много индекси, полагаме

$$\mathcal{L}_{\mathcal{A}}(q) \stackrel{\text{деф}}{=} \mathcal{L}(\mathcal{A}_q).$$

Нека да започнем с едно твърдение, което ни дава критерий, кога детерминизацията ни дава автомат с минимален брой състояния.

Твърдение 2.31. Нека $\mathcal{A}$ е НКА със следните свойства:

- За всяко състояние q е изпълнено, че:

$$\mathcal{L}_{\mathcal{A}}(q) \neq \emptyset. \tag{2.18}$$

- За всеки две състояния p и q е изпълнено, че:

$$p \neq q \implies \mathcal{L}_{\mathcal{A}}(q) \cap \mathcal{L}_{\mathcal{A}}(p) = \emptyset. \tag{2.19}$$

Тогава $\mathcal{M} = \texttt{det}(\mathcal{A})$, получен по метода на Рабин-Скот, е минимален автомат за езика на $\mathcal{A}$.

Критерият за минималност на детерминиран $\mathcal{A}$ гласи, че за всеки две различни състояния p и q, то $\mathcal{L}_{\mathcal{A}}(p) \neq \mathcal{L}_{\mathcal{A}}(q)$. Понеже тук $\mathcal{A}$ е недетерминиран, искаме по-силното свойство (2.19), а именно $\mathcal{L}_{\mathcal{A}}(p) \cap \mathcal{L}_{\mathcal{A}}(q) = \emptyset$. Съобразете, че това означава, че $\mathcal{A}$ има само едно финално състояние.

Упътване. От контрукцията на $\mathcal{M}$ в теоремата на Рабин-Скот, знаем, че всяко състояние на $\mathcal{M}$ е достижимо от началното. Критерият за минималност ни казва, че за да докажем, че $\mathcal{M}$ е минимален е достатъчно да покажем, че за произволни състояния P и U на автомата $\mathcal{M}$ е изпълнена импликацията:

$$\mathcal{L}_{\mathcal{M}}(P) = \mathcal{L}_{\mathcal{M}}(U) \implies P = U.$$

И така, да вземем две такива състояния P и U, за които $\mathcal{L}_{\mathcal{M}}(P) = \mathcal{L}_{\mathcal{M}}(U)$. Достатъчно е да докажем, че $P \subseteq U$, защото доказателството на другата посока е симетрично.

Да разгледаме произволен елемент $p \in P$. Щом от (2.18) имаме, че $\mathcal{L}_{\mathcal{A}}(p) \neq \emptyset$, то да разгледаме една дума $\omega \in \mathcal{L}_{\mathcal{A}}(p)$. Знаем, че $\Delta^{\star}_{\mathcal{A}}(\{p\}, \omega) \cap F^{\mathcal{A}} \neq \emptyset$ или с други думи, $\delta^{\star}_{\mathcal{M}}(P, \omega) \in F^{\mathcal{M}}$, откъдето имаме, че $\omega \in \mathcal{L}_{\mathcal{M}}(P)$. Сега, понеже приехме, че $\mathcal{L}_{\mathcal{M}}(P) = \mathcal{L}_{\mathcal{M}}(U)$, то $\delta^{\star}_{\mathcal{M}}(U, \omega) \in F^{\mathcal{M}}$, което означава, че за някое

състояние $u \in U$, $\Delta^\star_{\mathcal{A}}(\{u\}, \omega) \cap F^{\mathcal{A}} \neq \emptyset$. Така получаваме, че $\omega \in \mathcal{L}_{\mathcal{A}}(u)$. От (2.19) следва, че $u = p$, защото $\mathcal{L}_{\mathcal{A}}(p) \cap \mathcal{L}_{\mathcal{A}}(u) \neq \emptyset$. Заключаваме, че $P \subseteq U$. □

Твърдение 2.32. Нека $\mathcal{D}$ е ДКА, за който всяко състояние е достижимо от началното. Да разгледаме $\mathcal{A} \stackrel{\text{деф}}{=} \texttt{rev}(\mathcal{D})$. Тогава са изпълнени двете свойства от условието на *Твърдение 2.31*, а именно:

- За всяко състояние $q \in Q^{\mathcal{A}}$ е изпълнено, че:

$$\mathcal{L}_{\mathcal{A}}(q) \neq \emptyset.$$

- За всеки две състояния p и q е изпълнено, че:

$$p \neq q \implies \mathcal{L}_{\mathcal{A}}(q) \cap \mathcal{L}_{\mathcal{A}}(p) = \emptyset.$$

Упътване. Да напомним, че според *Задача 2.6*, $\mathcal{A} = (\Sigma, Q, \Delta, F, \{q_{\texttt{start}}\})$, където

$$\Delta_{\mathcal{A}}(q, a) \stackrel{\text{деф}}{=} \{p \in Q \mid \delta_{\mathcal{D}}(p, a) = q\}.$$

Щом в $\mathcal{D}$ всяко състояние q е достижимо от началното, то е ясно, че $\mathcal{L}_{\mathcal{A}}(q) \neq \emptyset$, което ни дава първото свойство. За второто свойство, нека приемем, че има дума $\omega \in \mathcal{L}_{\mathcal{A}}(p) \cap \mathcal{L}_{\mathcal{A}}(q)$. Ще докажем, че $p = q$. И така, щом $\omega \in \mathcal{L}_{\mathcal{A}}(p)$, то $\Delta^\star_{\mathcal{A}}(\{p\}, \omega) \ni q_{\texttt{start}}$, защото $q_{\texttt{start}}$ е единственото финално състояние на $\mathcal{A}$. Това означава, че $\delta^\star_{\mathcal{D}}(q_{\texttt{start}}, \omega) = p$. От друга страна, имаме също, че $\omega \in \mathcal{L}_{\mathcal{A}}(q)$, откъдето получаваме, че $\delta^\star_{\mathcal{D}}(q_{\texttt{start}}, \omega) = q$. Ясно е, че $p = q$, защото $\mathcal{D}$ е детерминиран автомат. □

Лема 2.9. Нека $\mathcal{D}$ е ДКА, за който всяко състояние е достижимо от началното. Тогава детерминираният краен автомат

$$\mathcal{M} \stackrel{\text{деф}}{=} \texttt{det}(\texttt{rev}(\mathcal{D}))$$

е минимален за езика $L^{\texttt{rev}}(\mathcal{D})$.

Упътване. Просто комбинираме горните две твърдения. Нека $\mathcal{A} \stackrel{\text{деф}}{=} \texttt{rev}(\mathcal{D})$. Знаем, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}^{\texttt{rev}}(\mathcal{D})$. От *Твърдение 2.32* знаем, че $\mathcal{A}$ притежава свойствата необходими за приложението на *Твърдение 2.31*. Така получаваме, че $\mathcal{M} \stackrel{\text{деф}}{=} \texttt{det}(\mathcal{A})$ е минимален детерминиран автомат за езика $\mathcal{L}(\mathcal{A})$. □

По този начин получаваме алгоритъм за минимизация на автомат.

Теорема 2.10 (Бжозовски)**.** Нека $\mathcal{A}$ е ДКА, за който всяко състояние е достижимо от началното. Тогава детерминираният краен автомат

$$\mathcal{M} \stackrel{\text{деф}}{=} \mathtt{det}(\mathtt{rev}(\mathtt{det}(\mathtt{rev}(\mathcal{A}))))$$

е минимален за езика на $\mathcal{A}$.

2.16 Хомоморфизми

Определение 2.8. За 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 е изпълнено, че $h(\varepsilon) = \varepsilon$. За да дефинирате един хомоморфизъм h е достатъчно да кажете как h преобразува всяка буква над азбуката Σ_1 в дума над азбуката Σ_2.

За произволна функция $h : \Sigma_1^\star \to \Sigma_2^\star$, да въведем означенията

- **Образът** на $L \subseteq \Sigma_1^\star$ относно h е множеството:

$$h(L) \stackrel{\text{деф}}{=} \{h(\omega) \in \Sigma_2^\star \mid \omega \in L\}.$$

- **Първообразът** на $L \subseteq \Sigma_2^\star$ относно h е множеството:

$$h^{-1}(L) \stackrel{\text{деф}}{=} \{\omega \in \Sigma_1^\star \mid h(\omega) \in L\}.$$

Твърдение 2.33. Нека $h : \Sigma_1^\star \to \Sigma_2^\star$ е функция. Тогава за всеки два езика $L_1 \subseteq \Sigma_1^\star$ и $L_2 \subseteq \Sigma_2^\star$ е изпълнено, че:

$$h(L_1 \cup L_2) = h(L_1) \cup h(L_2).$$

Доказателство. Достатъчно е да проследим следната верига от равенства:

Тук няма нужда h да бъде хомоморфизъм.

$$\begin{aligned} h(L_1 \cup L_2) &= \{h(\omega) \mid \omega \in L_1 \cup L_2\} \\ &= \{h(\omega) \mid \omega \in L_1 \vee \omega \in L_2\} \\ &= \{h(\omega) \mid \omega \in L_1\} \cup \{h(\omega) \mid \omega \in L_2\} \\ &= h(L_1) \cup h(L_2). \end{aligned}$$

□

Твърдение 2.34. Нека $h : \Sigma_1^\star \to \Sigma_2^\star$ е функция и нека за всяко n, L_n е език над азбуката Σ_1. Тогава

$$h(\bigcup_n L_n) = \bigcup_n h(L_n).$$

Доказателство. Просто проследяваме равенствата:

$$\begin{aligned} h(\bigcup_n L_n) &= \{h(\alpha) \mid (\exists n)[\alpha \in L_n]\} \\ &= \bigcup_n \{h(\alpha) \mid \alpha \in L_n\} \\ &= \bigcup_n h(L_n). \end{aligned}$$

□

Твърдение 2.35. Нека $h : \Sigma_1^\star \to \Sigma_2^\star$ е хомоморфизъм. Тогава за всеки два езика $L_1 \subseteq \Sigma_1^\star$ и $L_2 \subseteq \Sigma_2^\star$ е изпълнено, че:

$$h(L_1 \cdot L_2) = h(L_1) \cdot h(L_2).$$

Доказателство. Достатъчно е да проследим веригата от равенства:

$$\begin{aligned} h(L_1 \cdot L_2) &= \{h(\alpha) \mid \alpha \in L_1 \cdot L_2\} \\ &= \{h(\alpha_1\alpha_2) \mid \alpha_1 \in L_1 \,\&\, \alpha_2 \in L_2\} \\ &= \{h(\alpha_1) \cdot h(\alpha_2) \mid \alpha_1 \in L_1 \,\&\, \alpha_2 \in L_2\} \qquad \text{// } h \text{ е хомоморфизъм} \\ &= \{\omega_1\omega_2 \mid \omega_1 \in h(L_1) \,\&\, \omega_2 \in h(L_2)\} \\ &= h(L_1) \cdot h(L_2). \end{aligned}$$

□

Твърдение 2.36. Нека $h : \Sigma_1^\star \to \Sigma_2^\star$ е хомоморфизъм и $L \subseteq \Sigma_1^\star$. Тогава за всяко естествено число n е изпълнено, че:

$$h(L^n) = h(L)^n.$$

Доказателство. Индукция по n.

- Нека $n = 0$. Тогава всичко е ясно, защото:

$$L^0 = \{\varepsilon\} = h(L)^0.$$

- Да приемем като индукционно предположение, че $h(L^n) = h(L)^n$.

- За да докажем индукционната стъпка е достатъчно да проследим равенствата:

$$\begin{aligned} h(L^{n+1}) &= h(L^n \cdot L) \\ &= h(L^n) \cdot h(L) && \text{// от } \textit{Твърдение 2.35} \\ &= h(L)^n \cdot h(L) && \text{// от } \textbf{(И.П.)} \\ &= h(L)^{n+1}. \end{aligned}$$

□

Лема 2.10. Нека $h : \Sigma_1^\star \to \Sigma_2^\star$ е хомоморфизъм. Тогава, ако $L \subseteq \Sigma_1^\star$ е регулярен език, то $h(L)$ е регулярен език.

Лема 2.10 ни казва, че регулярните езици са затворени относно хомоморфизми.

Доказателство. Индукция по построението на регулярни езици.

- Нека $L = \emptyset$. Тогава е ясно, че

$$h(L) = \emptyset.$$

- Нека $L = \{a\}$, за някоя буква $a \in \Sigma$, то тогава

$$h(L) = \{h(a)\},$$

  който е ясно, че е регулярен.

- Нека L_1 и L_2 са регулярни езици, за които като индукционно предположение приемаме, че $h(L_1)$ и $h(L_2)$ са регулярни езици.

  – Според *Твърдение 2.33* имаме, че

$$h(L_1 \cup L_2) = h(L_1) \cup h(L_2).$$

  Щом от **(И.П.)** имаме, че $h(L_1)$ и $h(L_2)$ са регулярни езици, то заключаваме, че $h(L_1 \cup L_2)$ е регулярен език.

  – Според *Твърдение 2.35* имаме, че

$$h(L_1 \cdot L_2) = h(L_1) \cdot h(L_2).$$

  Щом от **(И.П.)** имаме, че $h(L_1)$ и $h(L_2)$ са регулярни езици, то заключаваме, че $h(L_1 \cdot L_2)$ е регулярен език.

- Нека сега L е регулярен език, за който да приемем като индукционно предположение, че $h(L)$ е регулярен език. Имаме равенствата:

$$\begin{aligned} h(L^\star) &= h(\bigcup_n L^n) && \text{// деф. на звезда на Клини} \\ &= \bigcup_n h(L^n) && \text{// от } \textit{Твърдение 2.34} \\ &= \bigcup_n h(L)^n && \text{// от } \textit{Твърдение 2.36} \\ &= h(L)^\star. && \text{// деф. на звезда на Клини} \end{aligned}$$

Щом от **(И.П.)** имаме, че $h(L)$ е регулярен език, то $h(L^\star) = h(L)^\star$ също е регулярен език.

□

Забележка. В общия случай, не можем да твърдим, че ако h е хомоморфизъм и $h(L)$ е регулярен, то L е регулярен. Нека за простота да фиксираме $\Sigma_1 = \Sigma_2 = \{a, b\}$.

Да вземем възможно най-простия хомоморфизъм h - този, който трие всичко, т.е. $h(a) \stackrel{\text{деф}}{=} \varepsilon$ и $h(b) \stackrel{\text{деф}}{=} \varepsilon$. Тогава за всеки език L, $h(L) = \{\varepsilon\}$.

Като друг пример, нека да вземем константен хомоморфизъм h - такъв, който винаги връща едно и също. Например, нека $h(a) \stackrel{\text{деф}}{=} a$ и $h(b) \stackrel{\text{деф}}{=} a$. За нерегулярния език $L = \{a^n b^n \mid n \in \mathbb{N}\}$ имаме, че $h(L) = \{a^k \mid k \text{ е четно}\}$, който е регулярен.

Пример 2.17. Да разгледаме езика $L = \{ca^n b^n \mid n \in \mathbb{N}\}$. Да видим как лесно можем да докажем, че L е нерегулярен. Да разгледаме хомоморфизма h, който изтрива буквата c, т.е. $h(a) \stackrel{\text{деф}}{=} a$, $h(b) \stackrel{\text{деф}}{=} b$ и $h(c) \stackrel{\text{деф}}{=} \varepsilon$. Тогава, ако допуснем, че L е регулярен, то би следвало, че и $h(L) = \{a^n b^n \mid n \in \mathbb{N}\}$ е регулярен. Достигхнаме до противоречие.

Лема 2.11. Нека $h : \Sigma_1^\star \to \Sigma_2^\star$ е хомоморфизъм. Тогава, ако $L \subseteq \Sigma_2^\star$ е регулярен език, то $h^{-1}(L)$ е регулярен език.

$h^{-1}(L)$ се нарича *първообразът* на L относно h.

Доказателство. Конструкция на автомат за $h^{-1}(L)$ при даден автомат за L. Нека $\mathcal{A}$ е ДКА разпознаващ езика L. Ще построим $\mathcal{A}_1 = \langle Q, \Sigma_1, \delta_1, q_{\texttt{start}}, F\rangle$, такъв че $\mathcal{L}(\mathcal{A}_1) = h^{-1}(L)$ като дефинираме функцията на преходите δ_1 като

$$\delta_1(q, a) \stackrel{\text{деф}}{=} \delta^\star(q, h(a)),$$

за произволно $q \in Q$ и $a \in \Sigma_1$. Важната стъпка в доказателството е да докажем, че:

$$(\forall \alpha \in \Sigma_1^\star)[\delta_1^\star(q, \alpha) = \delta^\star(q, h(\alpha))]. \tag{2.20}$$

- Нека $|\alpha| = 0$, т.е. $\alpha = \varepsilon$. Тогава директно получаваме, че

$$\delta_1^\star(q, \varepsilon) = \delta^\star(q, \varepsilon).$$

- Да приемем като индукционно предположение, че Свойство 2.20 е изпълнено за думи с дължина n.

- Нека сега разгледаме произволна дума α с дължина $n+1$, т.е. $\alpha = \beta a$, където $|\beta| = n$. Тогава:

$$\begin{aligned}
\delta_1^\star(q, \beta a) &= \delta_1(\delta_1^\star(q, \beta), a) && \text{// деф. на } \delta_1^\star \\
&= \delta_1(\delta^\star(q, h(\beta)), a) && \text{// } \textbf{(И.П.)} \\
&= \delta^\star(\delta^\star(q, h(\beta)), h(a)) && \text{// деф. на } \delta_1 \\
&= \delta^\star(q, h(\beta) \cdot h(a)) && \text{// } \textit{Твърдение 2.1} \\
&= \delta^\star(q, h(\beta a)). && \text{// } h \text{ е хомоморфизъм}
\end{aligned}$$

Сега лесно се вижда, че $h^{-1}(L) = \mathcal{L}(\mathcal{A}_1)$, защото:

$$\begin{aligned}
\alpha \in \mathcal{L}(\mathcal{A}_1) &\Leftrightarrow \delta_1^\star(q_{\texttt{start}}, \alpha) \in F \\
&\Leftrightarrow \delta^\star(q_{\texttt{start}}, h(\alpha)) \in F && \text{// Свойство 2.20} \\
&\Leftrightarrow h(\alpha) \in \mathcal{L}(\mathcal{A}) \\
&\Leftrightarrow \alpha \in h^{-1}(L). && \text{// } L = \mathcal{L}(\mathcal{A})
\end{aligned}$$

□

Забележка. Тук отново, в общия случай, не можем да твърдим, че ако h е хомоморфизъм и $h^{-1}(L)$ е регулярен, то L е регулярен. Нека за простота да фиксираме $\Sigma_1 = \Sigma_2 = \{a, b\}$.

Пример 2.18. Да видим защо езикът $L = \{a^n b a^n \mid n \in \mathbb{N}\}$ е нерегулярен. Да разгледаме хомоморфизма h_1 дефиниран като $h_1(a) \stackrel{\text{деф}}{=} a$, $h_1(b) \stackrel{\text{деф}}{=} ba$ и $h_1(c) \stackrel{\text{деф}}{=} a$. Тогава

Този пример е от [10, стр. 61].

$$h_1^{-1}(L) = \{\omega_1 b \omega_2 \mid |\omega_1| = |\omega_2| + 1 \ \& \ \omega_1, \omega_2 \in \{a, c\}^\star\}.$$

Да положим $L_1 = h^{-1}(L) \cap \mathcal{L}[\![\boldsymbol{a^\star b c^\star}]\!]$. Ясно е, че

$$L_1 = \{a^{n+1} b c^n \mid n \in \mathbb{N}\}.$$

Да разгледаме хомоморфизма h_2, където $h_2(a) \stackrel{\text{деф}}{=} a$, $h_2(b) \stackrel{\text{деф}}{=} c$, $h_2(c) \stackrel{\text{деф}}{=} c$. Тогава $h_2(L_1) = \{a^n c^n \mid n \geq 1\}$. Така получаваме, че ако L е регулярен, то езикът

$$h_2(h_1^{-1}(L) \cap \mathcal{L}[\![\boldsymbol{a^\star b c^\star}]\!])$$

също трябва да е регулярен, което е противоречие.

2.17 Допълнителни задачи

2.17.1 Лесни задачи

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

Задача 2.30. 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 2.31. 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) $L = \{\omega\omega^{\mathtt{rev}} \mid \omega \in \{a, b\}^\star\}$;

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 2.32. 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 2.33. 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 2.34. Нека L_1 и L_2 са регулярни езици. Докажете, че L също е регулярен език, където

$$L = \{\alpha \mid (\exists \beta, \gamma)[\beta\alpha\gamma \in L_1] \;\&\; \alpha \in L_2 \vee \alpha^{\mathtt{rev}} \in L_2\}.$$

Задача 2.35. 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

$$L_1 \oplus L_2 = \{\omega \in (\Sigma_1 \cup \Sigma_2)^\star \mid \omega_1 \in L_1 \;\&\; \omega_2 \in L_2 \;\&\; |\omega_1| = |\omega_2|\}.$$

а) Вярно ли е, че ако L_1 е краен, то $L_1 \oplus L_2$ е регулярен език? Да

б) Вярно ли е, че ако L_1 и L_2 са регулярни езици, то $L_1 \oplus L_2$ е регулярен език? Не

Задача 2.36. 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

$$L_1 \oplus L_2 = \{\omega \in (\Sigma_1 \cup \Sigma_2)^\star \mid \omega_1 \in L_1 \;\&\; \omega_2 \in L_2\}.$$

Вярно ли е, че ако L_1 и L_2 са регулярни езици, то $L_1 \oplus L_2$ е регулярен език? Да

Задача 2.37. Нека $\Sigma = \{a, b, c\}$ и да разгледаме следната операция [7, стр. 88].

$$\mathtt{Sort} : \mathcal{P}(\Sigma^\star) \to \mathcal{P}(\Sigma^\star),$$

където

$$\mathtt{Sort}(L) = \{a^{|\omega|_a} b^{|\omega|_b} c^{|\omega|_c} \mid \omega \in L\}.$$

Вярно ли е, че ако L е регулярен, то $\mathtt{Sort}(L)$ е регулярен ?

2.17.2 Не толкова лесни задачи

[18, стр. 84]

Задача 2.38. При дадени езици L и M над азбуката Σ, да разгледаме:

а) $\texttt{NoExtend}(L) = \{\alpha \in L \mid \alpha$ не е префикс на никоя дума от $L\}$;

б) $\text{Suf}(L) = \{\beta \in \Sigma^\star \mid (\exists \alpha \in \Sigma^\star)[\alpha\beta \in L]\}$;

в) $\text{Infix}(L) = \{\alpha \mid (\exists \beta, \gamma \in \Sigma^\star)[\beta\alpha\gamma \in L]\}$;

right quotient of L by ω

г) $L/_\omega = \{\alpha \in \Sigma^\star \mid \alpha\omega \in L\}$;

д) $L/_M = \{\alpha \in \Sigma^\star \mid (\exists \beta \in M)[\alpha\beta \in L]\}$;

е) $\text{Max}(L) = \{\alpha \in \Sigma^\star \mid (\forall \beta \in \Sigma^\star)[\beta \neq \varepsilon \implies \alpha\beta \notin L]\}$.

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

[13, стр. 75]; [18, стр. 89]

Задача 2.39. Да фиксираме азбука само с един символ $\Sigma = \{a\}$. Да положим за всяко $p, q \in \mathbb{N}$,

$$L(p, q) = \{a^k \mid (\exists n \in \mathbb{N})[k = p + q \cdot n]\}.$$

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 2.40. 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 2.41. 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) $\frac{2}{3}(L) = \{\beta \in \Sigma^\star \mid (\exists \alpha, \gamma \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \ \& \ \alpha\beta\gamma \in L]\}$;

д) $\frac{3}{3}(L) = \{\gamma \in \Sigma^\star \mid (\exists \alpha, \beta \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \ \& \ \alpha\beta\gamma \in L]\}$;

е) $\hat{L} = \{\alpha\gamma \in \Sigma^\star \mid (\exists \beta, \gamma \in \Sigma^\star)[|\alpha| = |\beta| = |\gamma| \ \& \ \alpha\beta\gamma \in L]\}$;

Проверете ако L е регулярен, то кои от горните езици също са регулярни.

Задача 2.42. 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 2.43. Да фиксираме една азбука Σ. Да дефинираме функция $\texttt{diff} : \Sigma^\star \times \Sigma^\star \to \mathbb{N}$, която брои колко разлики има между α и β по следния начин:

$$\begin{aligned}
\texttt{diff}(\varepsilon, \beta) &\stackrel{\text{деф}}{=} |\beta| \\
\texttt{diff}(\alpha, \varepsilon) &\stackrel{\text{деф}}{=} |\alpha| \\
\texttt{diff}(x\alpha, y\beta) &\stackrel{\text{деф}}{=} \begin{cases} 1 + \texttt{diff}(\alpha, \beta), & \text{ако } x \neq y \\ \texttt{diff}(\alpha, \beta), & \text{иначе.} \end{cases}
\end{aligned}$$

За произволни езици L и M, да дефинираме езика

$$\texttt{Diff}_n(L, M) \stackrel{\text{деф}}{=} \{\alpha \in L \mid (\exists \beta \in M)[\texttt{diff}(\alpha, \beta) = n]\}.$$

Докажете, че ако L и M са регулярни, то $\texttt{Diff}_n(L, M)$ е регулярен.

Задача 2.44. 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.

Да обърнем внимание, че езикът $\{\alpha\sharp\beta\sharp\gamma \mid \overline{\alpha}_{(2)} + \overline{\beta}_{(2)} = \overline{\gamma}_{(2)}\}$ не е регулярен.

Упътване. Доста по-удобно е да построим автомат $\mathcal{A}$, такъв че $\mathcal{L}(\mathcal{A}) = L^{\mathtt{rev}}$. Да започнем с състоянието $q_=$, за което искаме да имаме свойството, че за произволно състояние q,

$$\delta^\star(q, \begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix}) = q_= \Leftrightarrow \overline{\alpha^{\mathtt{rev}}}_{(2)} + \overline{\beta^{\mathtt{rev}}}_{(2)} = \overline{\gamma^{\mathtt{rev}}}_{(2)}.$$

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:

$$\begin{aligned}
&\delta(q_=, \begin{bmatrix}0\\0\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_= && // \ \overline{0\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{0\gamma}_{(2)}\\
&\delta(q_=, \begin{bmatrix}0\\1\\1\end{bmatrix}) \stackrel{\text{деф}}{=} q_= && // \ \overline{0\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{1\gamma}_{(2)}\\
&\delta(q_=, \begin{bmatrix}1\\0\\1\end{bmatrix}) = q_= && // \ \overline{1\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{1\gamma}_{(2)}
\end{aligned}$$

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

$$\delta(q_=, \gamma) \stackrel{\text{деф}}{=} q_{\mathtt{err}},$$

където $q_{\mathtt{err}}$ 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,

$$\delta^\star(q, \begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix}) = q_1 \Leftrightarrow \overline{\alpha^{\mathtt{rev}}}_{(2)} + \overline{\beta^{\mathtt{rev}}}_{(2)} = \overline{1\gamma^{\mathtt{rev}}}_{(2)}.$$

Да разгледаме сега случая $\overline{\alpha}_{(2)} + \overline{\beta}_{(2)} = \overline{1\gamma}_{(2)}$. Тогава:

$$\begin{aligned}
&\delta(q_1, \begin{bmatrix}0\\0\\1\end{bmatrix}) \stackrel{\text{деф}}{=} q_= && // \ \overline{0\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{1\gamma}_{(2)}\\
&\delta(q_1, \begin{bmatrix}1\\1\\1\end{bmatrix}) \stackrel{\text{деф}}{=} q_1 && // \ \overline{1\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{11\gamma}_{(2)}\\
&\delta(q_1, \begin{bmatrix}1\\0\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_1 && // \ \overline{1\alpha}_{(2)} + \overline{0\beta}_{(2)} = \overline{10\gamma}_{(2)}\\
&\delta(q_1, \begin{bmatrix}0\\1\\0\end{bmatrix}) \stackrel{\text{деф}}{=} q_1 && // \ \overline{0\alpha}_{(2)} + \overline{1\beta}_{(2)} = \overline{10\gamma}_{(2)}\\
&\delta(q_1, \gamma) \stackrel{\text{деф}}{=} q_{\mathtt{err}} && // \ \text{за останалите } \gamma \in \Sigma_3
\end{aligned}$$

□

Задача 2.45. 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:

- $L_1 = \left\{ \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \in \Sigma_2^\star \mid \overline{\alpha}_{(2)} < \overline{\beta}_{(2)} \right\}$;
- $L_2 = \left\{ \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \in \Sigma_2^\star \mid 3(\overline{\alpha}_{(2)}) = \overline{\beta}_{(2)} \right\}$;
- $L_3 = \left\{ \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \in \Sigma_2^\star \mid \alpha = \beta^{\mathtt{rev}} \right\}$;

Докажете, че L_1 и L_2 са автоматни, а L_3 не е автоматен.

Упътване. Ще построим автомат $\mathcal{A} = \langle \Sigma, Q, q_{\mathtt{start}}, \delta, F \rangle$ за езика $L_1^{\mathtt{rev}}$. За улеснение, в рамките на тази задача ще пишем:

- $\alpha \equiv \beta$, ако $\overline{\alpha^{\mathtt{rev}}}_{(2)} = \overline{\beta^{\mathtt{rev}}}_{(2)}$,
- $\alpha \prec \beta$, ако $\overline{\alpha^{\mathtt{rev}}}_{(2)} < \overline{\beta^{\mathtt{rev}}}_{(2)}$,
- $\alpha \succ \beta$, ако $\overline{\alpha^{\mathtt{rev}}}_{(2)} > \overline{\beta^{\mathtt{rev}}}_{(2)}$.

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 $q_{\mathtt{start}} = q_=$. 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 $\mathcal{A}$ е изпълнено, че $\mathcal{L}(\mathcal{A}) = L_1^{\mathtt{rev}}$. □

Нека $\Sigma^{\geq k} \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid |\alpha| \geq k\}$.

Задача 2.46. Нека L е регулярен език. Докажете, че езиците

- $L_1 = \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha[\!: i] \in L\,]\}$;
- $L_2 = \{\alpha \in \Sigma^\star \mid (\forall i)[\, \alpha[\!: i] \in L\,]\}$;

също са регулярни.

Упътване. Първо, лесно се вижда, че $L_1 = L \cdot \Sigma^\star$. Второ, съобразете, че

$$\begin{aligned}\overline{L}_2 &= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha[\!: i] \in \overline{L}\,]\}\\ &= \overline{L} \cdot \Sigma^\star.\end{aligned}$$

Оттук заключете, че $L_2 = \overline{\overline{L} \cdot \Sigma^\star}$. □

Задача 2.47. Нека L е регулярен език. Тогава езиците

- $L_1 = \{\alpha \in \Sigma^\star \mid (\exists i)(\exists j)[\, i < j \;\&\; \alpha[i : j] \in L\,]\}$;
- $L_2 = \{\alpha \in \Sigma^\star \mid (\forall i)(\forall j)[\, i < j \implies \alpha[i : j] \in L\,]\}$;
- $L_3 = \{\alpha \in \Sigma^\star \mid (\forall i)(\exists j)[\, i < j \;\&\; \alpha[i : j] \in L\,]\}$;
- $L_4 = \{\alpha \in \Sigma^\star \mid (\exists i)(\forall j)[\, i < j \implies \alpha[i : j] \in L\,]\}$.

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

$$\begin{aligned}\alpha \in \Sigma^\star \cdot L &\Leftrightarrow (\exists \beta \in \Sigma^\star)(\exists \gamma \in L)[\alpha = \beta \cdot \gamma]\\ &\Leftrightarrow (\exists j)[\, \alpha[j :] \in L\,].\end{aligned}$$

Аналогично получаваме, че

$$\begin{aligned}\alpha \in L \cdot \Sigma^\star &\Leftrightarrow (\exists \gamma \in L)(\exists \beta \in \Sigma^\star)[\alpha = \gamma \cdot \beta]\\ &\Leftrightarrow (\exists j)[\, \alpha[\!: j] \in L\,].\end{aligned}$$

Нека да разгледаме по-подробно следния език:

$$\begin{aligned}\overline{L}_3 &= \{\alpha \in \Sigma^\star \mid (\exists i)(\forall j)[\, i < j \implies \alpha[i : j] \notin L\,]\}\\ &= \{\alpha \in \Sigma^\star \mid (\exists i)(\forall j)[\, i < j \implies \alpha[i : j] \in \overline{L}\,]\}\\ &= \{\alpha \in \Sigma^\star \mid (\forall j)[\, \alpha[\!: j] \in \Sigma^\star \cdot \overline{L}\,]\}\end{aligned}$$

Така получаваме, че:

$$\begin{aligned}L_3 &= \{\alpha \in \Sigma^\star \mid (\exists j)[\, \alpha[\!: j] \notin \Sigma^\star \cdot \overline{L}\,]\}\\ &= \{\alpha \in \Sigma^\star \mid (\exists j)[\, \alpha[\!: j] \in \overline{\Sigma^\star \cdot \overline{L}}\,]\}\\ &= \{\alpha \in \Sigma^\star \mid \alpha \in (\overline{\Sigma^\star \cdot \overline{L}}) \cdot \Sigma^\star\,]\}.\end{aligned}$$

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 2.48. Нека L е регулярен език над азбуката Σ. За произволно естествено число k, докажете, че езикът

$$L_k = \{\alpha \in \Sigma^\star \mid (\forall i)(\exists j)[\, |\alpha[i:]| \geq k \implies \alpha[i:j] \in L\,]\}$$

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.

$$\begin{aligned}
L_k &= \{\alpha \in \Sigma^\star \mid (\forall i)(\exists j)[\, |\alpha[i:]| \geq k \implies \alpha[i:j] \in L\,]\} \\
&= \{\alpha \in \Sigma^\star \mid (\forall i)[\, |\alpha[i:]| \geq k \implies (\exists j)[\alpha[i:j] \in L]\,]\} \\
&= \{\alpha \in \Sigma^\star \mid (\forall i)[\, \alpha[i:] \in \Sigma^{\geq k} \implies \alpha[i:] \in L \cdot \Sigma^\star\,]\} \\
&= \{\alpha \in \Sigma^\star \mid (\forall i)[\, \alpha[i:] \notin \Sigma^{\geq k} \vee \alpha[i:] \in L \cdot \Sigma^\star\,]\} \\
&= \{\alpha \in \Sigma^\star \mid \neg(\exists i)[\, \alpha[i:] \in \Sigma^{\geq k} \,\&\, \alpha[i:] \notin L \cdot \Sigma^\star\,]\}.
\end{aligned}$$

Сега е ясно, че:

$$\begin{aligned}
\overline{L}_k &= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha[i:] \in \Sigma^{\geq k} \,\&\, \alpha[i:] \notin L \cdot \Sigma^\star\,]\} \\
&= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha[i:] \in \Sigma^{\geq k} \,\&\, \alpha[i:] \in \overline{L \cdot \Sigma^\star}\,]\} \\
&= \{\alpha \in \Sigma^\star \mid (\exists i)[\, \alpha[i:] \in \Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star}\,]\} \\
&= \{\alpha \in \Sigma^\star \mid \alpha \in \Sigma^\star \cdot (\Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star})\} \\
&= \Sigma^\star \cdot (\Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star}).
\end{aligned}$$

Тогава

$$L_k = \overline{\Sigma^\star \cdot (\Sigma^{\geq k} \cap \overline{L \cdot \Sigma^\star})}.$$

□

Нека $\mathcal{A} = \langle \Sigma, Q, q_{\texttt{start}}, \delta, F \rangle$. За някои по-сложни задачи е удобно да въведем означението

$$\mathcal{L}_{\mathcal{A}}(q, P) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \delta^\star_{\mathcal{A}}(q, \omega) \in P\},$$

където $q \in Q$ и $P \subseteq Q$. В частност, $\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}}, F)$. Лесно се съобразява, че:

$$\mathcal{L}(\mathcal{A}) = \bigcup_{p \in Q} \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}}, \{p\}) \cdot \mathcal{L}_{\mathcal{A}}(p, F).$$

Аналогично, ако $\mathcal{A} = \langle \Sigma, Q, Q_{\texttt{start}}, \Delta, F \rangle$ е НКА, то да положим

$$\mathcal{L}_{\mathcal{A}}(q, P) \stackrel{\text{деф}}{=} \{\omega \in \Sigma^\star \mid \Delta^\star_{\mathcal{A}}(\{q\}, \omega) \cap P \neq \emptyset\}.$$

Циклични разместявания

Да разгледаме следната операция върху езици

$$\texttt{Cycle}(L) \stackrel{\text{деф}}{=} \{\omega_2 \cdot \omega_1 \mid \omega_1 \cdot \omega_2 \in L\}.$$

Задача 2.49. Докажете, че ако L е регулярен, то $\texttt{Cycle}(L)$ е регулярен.

В [24, стр. 60] е дадено конструктивно решение на тази задача, т.е. в явен вид се строи автомат. Нашият подход е алгебричен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Достатъчно е да проследим веригата от еквивалентности:

$$\begin{aligned}
\omega \in \texttt{Cycle}(L) &\Leftrightarrow (\exists i)[\, \omega\texttt{[}i\,\texttt{:]} \cdot \omega\texttt{[:}\,i\texttt{]} \in L \,] \\
&\Leftrightarrow (\exists i)(\exists p \in Q)[\, \omega\texttt{[}i\,\texttt{:]} \in \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}}, \{p\}) \;\&\; \omega\texttt{[:}\,i\texttt{]} \in \mathcal{L}_{\mathcal{A}}(p, F) \,] \\
&\Leftrightarrow (\exists p \in Q)[\, \omega\texttt{[:}\,i\texttt{]} \cdot \omega\texttt{[}i\,\texttt{:]} \in \mathcal{L}_{\mathcal{A}}(p, F) \cdot \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}}, \{p\}) \,] \\
&\Leftrightarrow (\exists p \in Q)[\, \omega \in \mathcal{L}_{\mathcal{A}}(p, F) \cdot \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}}, \{p\}) \,] \\
&\Leftrightarrow \omega \in \bigcup_{p \in Q} \mathcal{L}_{\mathcal{A}}(p, F) \cdot \mathcal{L}_{\mathcal{A}}(q_{\texttt{start}}, \{p\}).
\end{aligned}$$

□

Разполовяване и удвояване на думи

Нека да видим сега следните две операции върху езици:

$$\texttt{Half}(L) \stackrel{\text{деф}}{=} \{\omega \mid \omega \cdot \omega \in L\},$$
$$\texttt{Double}(L) \stackrel{\text{деф}}{=} \{\omega \cdot \omega \mid \omega \in L\}.$$

На пръв поглед може би изглежда, че тези две операции имат еднакви свойства относно регулярните езици. Оказва се, че това не е така.

Задача 2.50. Докажете, че ако L е регулярен, то $\texttt{Half}(L)$ е регулярен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Достатъчно е да проследим веригата от еквивалентности:

$$\begin{aligned}\omega \in \texttt{Half}(L) &\Leftrightarrow \omega \cdot \omega \in L \\ &\Leftrightarrow (\exists p \in Q)[\delta^\star_\mathcal{A}(q_\texttt{start}, \omega) = p \;\&\; \delta^\star_\mathcal{A}(p, \omega) \in F] \\ &\Leftrightarrow (\exists p \in Q)[\omega \in \mathcal{L}_\mathcal{A}(q_\texttt{start}, \{p\}) \;\&\; \omega \in \mathcal{L}_\mathcal{A}(p, F)] \\ &\Leftrightarrow \omega \in \bigcup_{p \in Q} (\mathcal{L}_\mathcal{A}(q_\texttt{start}, \{p\}) \cap \mathcal{L}_\mathcal{A}(p, F)).\end{aligned}$$

Ясно е, че:

$$\texttt{Half}(L) = \bigcup_{p \in Q} (\mathcal{L}_\mathcal{A}(q_\texttt{start}, \{p\}) \cap \mathcal{L}_\mathcal{A}(p, F)).$$

Така изразихме $\texttt{Half}(L)$ като крайна булева комбинация на регулярни езици. Следователно, $\texttt{Half}(L)$ е регулярен език. □

Задача 2.51. Съществува регулярен език L, за който $\texttt{Double}(L)$ не е регулярен.

Упътване. Достатъчно е да разгледаме $L = \mathcal{L}[\![\boldsymbol{a^\star b}]\!]$. Тогава

$$\texttt{Double}(L) = \{a^n b a^n b \mid n \in \mathbb{N}\},$$

за който лесно се съобразява, че не е регулярен. □

Триене на половината дума

Да разгледаме следната операция върху езици:

$$\frac{1}{2}(L) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid (\exists \beta)[\, |\alpha| = |\beta| \;\&\; \alpha\beta \in L\,]\}.$$

Задача 2.52. Докажете, че ако L е регулярен, то $\frac{1}{2}(L)$ е регулярен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Да вземем НКА $\mathcal{B}$, който е същия като $\mathcal{A}$, но всеки преход в $\mathcal{A}$ се замества с преход с произволна буква.

В [24, стр. 58] в явен вид се строи автомат.

- $Q_{\mathcal{B}} = Q_{\mathcal{A}}$;
- $Q^{\mathcal{B}}_{\mathtt{start}} = \{q^{\mathcal{A}}_{\mathtt{start}}\}$;
- $F_{\mathcal{B}} = F_{\mathcal{A}}$;
- $\Delta_{\mathcal{B}}(q, x) = \{\delta_{\mathcal{A}}(q, y) \mid y \in \Sigma\}$.

Тогава $\beta \in \mathcal{L}(\mathcal{B}) \Leftrightarrow (\exists\alpha)[|\alpha| = |\beta| \,\&\, \alpha \in \mathcal{L}(\mathcal{A})]$.

$$\begin{aligned}\alpha \in \mathcal{L}(\mathcal{A}) &\Leftrightarrow (\exists p \in Q)[\alpha \in \mathcal{L}_{\mathcal{A}}(q_{\mathtt{start}}, \{p\}) \,\&\, \alpha \in \mathcal{L}_{\mathcal{B}}(\{p\}, F)] \\ &\Leftrightarrow \alpha \in \bigcup_{p \in Q} (\mathcal{L}_{\mathcal{A}}(q_{\mathtt{start}}, \{p\}) \cap \mathcal{L}_{\mathcal{B}}(p, F)).\end{aligned}$$

□

Триене на третина от дума

Да разгледаме следните операции върху езици.

$$\mathtt{Triples}_2(L) \stackrel{\text{деф}}{=} \{\omega_2 \mid (\exists\omega_1)(\exists\omega_3)[\, \omega_1\omega_2\omega_3 \in L \,\&\, |\omega_1| = |\omega_2| = |\omega_3| \,]\,\};$$
$$\mathtt{Triples}_{1,3}(L) \stackrel{\text{деф}}{=} \{\omega_1\omega_3 \mid (\exists\omega_2)[\omega_1\omega_2\omega_3 \in L \,\&\, |\omega_1| = |\omega_2| = |\omega_3|]\}.$$

Отново се оказва, че тези две операции са принципно различни.

Задача 2.53. Докажете, че ако L е регулярен език, то $\mathtt{Triples}_2(L)$ е регулярен.

Интересно е, че $\mathtt{Triples}_{1,3}(L)$ е безконтекстен *Задача 3.45*.

Задача 2.54. Докажете, че съществува регулярен език L, за който $\mathtt{Triples}_{1,3}(L)$ не е регулярен.

Упътване. Разгледайте езика $L = \mathcal{L}[\![\boldsymbol{a}^\star\sharp\boldsymbol{a}^+\sharp\boldsymbol{a}^\star]\!]$. Ако допуснем, че $\mathtt{Triples}_{1,3}(L)$ е регулярен, то

$$\mathtt{Triples}_{1,3}(L) \cap \mathcal{L}[\![\boldsymbol{a}^\star\sharp\sharp\boldsymbol{a}^\star]\!] = \{a^n\sharp\sharp a^n \mid n \in \mathbb{N}\}$$

също би трябвало да е регулярен, но лесно се съобразява, че $\{a^n\sharp\sharp a^n \mid n \in \mathbb{N}\}$ не е регулярен. □

Триене на дължината на квадрат

Да разгледаме следната операция върху езици:

$$\sqrt{L} \stackrel{\text{деф}}{=} \{\alpha \mid (\exists\beta)[\, |\beta| = |\alpha|^2 \,\&\, \alpha \cdot \beta \in L \,]\}.$$

Задача 2.55. Докажете, че ако L е регулярен, то $\sqrt{L}$ е регулярен.

Триене на степен на дължината

Да разгледаме следната операция върху езици:

$$\log(L) \stackrel{\text{деф}}{=} \{\alpha \mid (\exists\beta)[\, |\beta| = 2^{|\alpha|} \,\&\, \alpha\cdot\beta \in L\,]\}.$$

Задача 2.56. Докажете, че ако L е регулярен, то $\log(L)$ е регулярен.

Смесване на думи

Да дефинираме рекурсивно функцията $\texttt{merge} : \Sigma^\star \times \Sigma^\star \to \mathcal{P}(\Sigma^\star)$, която смесва две думи по всички възможни начини така:

Тук следваме [24, стр. 57]. Това е добра задача, при която е удачно да се използват хомоморфизми.

$$\texttt{merge}(\alpha, \varepsilon) \stackrel{\text{деф}}{=} \{\alpha\};$$
$$\texttt{merge}(\varepsilon, \beta) \stackrel{\text{деф}}{=} \{\beta\};$$
$$\texttt{merge}(a\cdot\alpha, b\cdot\beta) \stackrel{\text{деф}}{=} \{a\}\cdot\texttt{merge}(\alpha, b\cdot\beta) \cup \{b\}\cdot\texttt{merge}(a\cdot\alpha, \beta).$$

Например,

$$\texttt{merge}(ab, cd) = \{abcd, acbd, acdb, cabd, cadb, cdab\}.$$

Сега можем да дефинираме смесването на два езика

$$\texttt{Merge}(L_1, L_2) \stackrel{\text{деф}}{=} \bigcup_{\alpha\in L_1, \beta\in L_2} \texttt{merge}(\alpha, \beta).$$

Задача 2.57. Ако L_1 и L_2 са регулярни езици над азбуката Σ, то $\texttt{Merge}(L_1, L_2)$ е регулярен език.

Упътване. Нека имаме азбука $\Sigma' \stackrel{\text{деф}}{=} \{x' \mid x \in \Sigma\}$. Дефинираме хомоморфизъм h_1 като:

$$h_1(x) \stackrel{\text{деф}}{=} x$$
$$h_1(x') \stackrel{\text{деф}}{=} \varepsilon.$$

Така получаваме, че $h_1^{-1}(a) = (\Sigma')^\star \cdot \{a\} \cdot (\Sigma')^\star$.

Дефинираме хомоморфизъм h_2 като:

$$h_2(x) \stackrel{\text{деф}}{=} \varepsilon$$
$$h_2(x') \stackrel{\text{деф}}{=} x.$$

Аналогично, $h_2^{-1}(a) = \Sigma^\star \cdot \{a'\} \cdot \Sigma^\star$. Оттук следва, че

$$h_1^{-1}(a) \cap h_2^{-1}(b) = \{a'b, ab'\}.$$

Накрая дефинираме хомоморфизма h като

$$h(x) \stackrel{\text{деф}}{=} x$$
$$h(x') \stackrel{\text{деф}}{=} x.$$

Така получаваме, че

$$\texttt{Merge}(L_1, L_2) = h(h_1^{-1}(L_1) \cap h_2^{-1}(L_2)).$$

□

Префикси

Нека положим

$$\texttt{Prefix}(L) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid (\exists \beta \in L)[\alpha \preceq \beta]\},$$
$$\texttt{PrefixFree}(L) \stackrel{\text{деф}}{=} \{\alpha \in L \mid \neg(\exists \beta \in L)[\beta \prec \alpha]\}.$$

Задача 2.58. Докажете, че ако L е регулярен език, то $\texttt{Prefix}(L)$ също е регулярен.

Упътване. Възможно е да докажем това твърдение с конструкция върху автомат $\mathcal{A}$ разпознаващ L. Възможно е и да докажем твърдението с индукция по построението на регулярните езици. □

Задача 2.59. Винаги ли е вярно, че ако $\texttt{Prefix}(L)$ е регулярен, то L е регулярен ?

Упътване. Не. Възможно е езикът L да не е регулярен, но $\texttt{Prefix}(L)$ да е регулярен. Например, разгледайте езика $L \stackrel{\text{деф}}{=} \{a^{n^2} \mid n \in \mathbb{N}\}$ и съобразете, че $\texttt{Prefix}(L) = \{a\}^\star$. □

Задача 2.60. Докажете, че съществуват безкрайни езици L, за които

$$L = \texttt{PrefixFree}(L).$$

Упътване. Съобразете, че за езика $L \stackrel{\text{деф}}{=} \{a\}^\star \cdot \{b\}$ е изпълнено, че $L = \texttt{PrefixFree}(L)$. □

Задача 2.61. Докажете, че ако L е регулярен, то $\texttt{PrefixFree}(L)$ е регулярен.

Упътване. Съобразете, че $\texttt{PrefixFree}(L) = L \setminus L \cdot \Sigma^+$. □

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 Синтактични дървета

> A parse tree is a graphical representation of a derivation that filters out the order in which productions are applied to replace nonterminals. [1]

На английски се нарича parse tree, syntax tree, derivation tree. Обикновено, например в [18, стр. 123], дават рекурсивна дефиниция на синтактично дърво. Тук използваме дефиницията на понятието дърво разгледана в Раздел 1.7. Функцията λ облича голото дърво T с елементи на граматиката G.

- 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

    $$X_\alpha \to_G X_{\alpha\cdot 0} X_{\alpha\cdot 1} \cdots X_{\alpha\cdot k},$$

    където $X_{\alpha i} \in V \cup \Sigma$.

    С други думи, $\lambda(\alpha) \to_G \lambda(\alpha\cdot 0) \cdots \lambda(\alpha\cdot k)$.

    Това свойство е ключово, защото то ни казва, че дървото е съвместимо с правилата на граматиката G.
  - Имаме частен случай за $\alpha \in T$ и $|\mathtt{succ}_T(\alpha)| = 1$. Тогава позволяваме да имаме $X_{\alpha\cdot 0} = \varepsilon$, 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

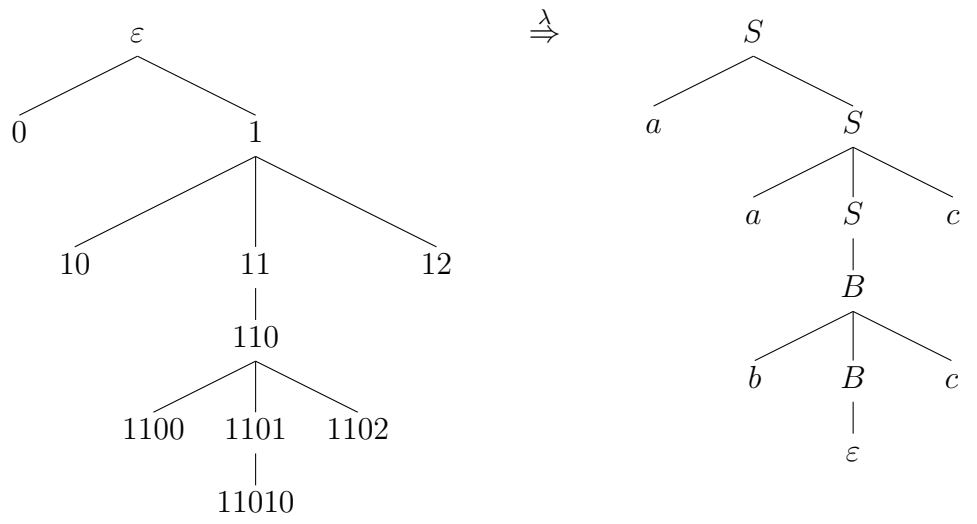
$$\mathtt{yield}(P) \stackrel{\text{деф}}{=} \lambda(\alpha_0) \cdot \lambda(\alpha_1) \cdots \lambda(\alpha_k).$$

Пример 3.1. 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 3.1: 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.

3.2 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:

$$\frac{X \in V \cup \Sigma}{X \overset{0}{\lhd} X} \text{ правило (0)}$$

$$(\ell = \sup\{\ell_1, \ldots, \ell_n\}) \quad \frac{X \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \gamma_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \gamma_n}{X \overset{\ell+1}{\lhd} \gamma_1 \cdots \gamma_n} \text{ правило (1)}$$

Според дефиницията $\sup(A) \overset{\text{деф}}{=} \min\{m \in \mathbb{N} \mid (\forall a \in A)[a \leq m]\}$ е ясно, че $\sup(\emptyset) = 0$.

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 $A \overset{1}{\lhd} \varepsilon$. Друг важен случай е когато $X \in \Sigma$. Тогава имаме, че $X \overset{0}{\lhd} X$, но *нямаме* $X \overset{\ell}{\lhd} X$ за всяко $\ell > 0$.

Определение 3.2. Нека G е произволна безконтекстна граматика. Тогава дефинираме езикът на G да бъде

$$\mathcal{L}(G) \overset{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid S \overset{\star}{\lhd} \alpha\}.$$

Естествено е да разширим релацията $\overset{\ell}{\lhd}$ до бинарна релация над $(V \cup \Sigma)^\star$ следния начин:

$$(\ell = \sup\{\ell_1, \ldots, \ell_n\}) \quad \frac{X_1 \overset{\ell_1}{\lhd} \alpha_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \alpha_n}{X_1 \cdots X_n \overset{\ell}{\lhd} \alpha_1 \cdots \alpha_n}$$

Тук, отново, ако $n = 0$, то $\ell = 0$ и $X_1 \cdots X_n = \varepsilon$. Така получаваме извода $\varepsilon \overset{0}{\lhd} \varepsilon$.

Да дефинираме релацията $\overset{\star}{\lhd}$ по следния начин:

$$\alpha \overset{\star}{\lhd} \beta \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[\alpha \overset{\ell}{\lhd} \beta].$$

Съобразете, че имаме:

$$\frac{X \to_G \gamma}{X \overset{1}{\lhd} \gamma.}$$

Задача 3.1. Докажете, че:

$$\frac{\alpha \overset{\ell_1}{\lhd} \beta \qquad \beta \overset{\ell_2}{\lhd} \gamma}{\alpha \overset{\leq \ell_1+\ell_2}{\lhd} \gamma}$$

Релацията $\overset{\star}{\lhd}$ е удобна, когато не се интересуваме от реда, по който прилагаме правилата за извод в една безконтекстна граматика.

Следващото твърдение ни казва, че има пряка връзка между релацията $\overset{\star}{\lhd}$ и синтактичните дървета.

Твърдение 3.1. 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.

- Нека $\ell = 0$. Тогава твърдението е ясно.
- Нека $\ell > 0$. Първо, нека $X \overset{\ell}{\lhd} \gamma$. От правилата за извод върху синтактично дърво следва, че съществува правило $X \to_G X_0 \cdots X_{n-1}$ и $X_i \overset{\ell_i}{\lhd} \gamma_i$ за $i < n$, като $\ell = \sup\{\ell_0, \ldots, \ell_{n-1}\} + 1$ и $\gamma = \gamma_0 \cdots \gamma_{n-1}$. От **(И.П.)** , съществуват синтактични дървета P_i, за които $\texttt{root}(P_i) = X_i$, $\texttt{yield}(P_i) = \gamma_i$ и $\texttt{height}(P_i) = \ell_i$. Така можем да направим ново дърво $P = (T, \lambda)$, за което $T = \{\varepsilon\} \cup \bigcup_{i<n}\{i\} \cdot T_i$ и $\lambda(\varepsilon) = X$, $\lambda(i\alpha) = \lambda_i(\alpha)$.

  Второ, нека $P = (T, \lambda)$ е синтактично дърво, за което $\texttt{root}(P) = X$, $\texttt{yield}(P) = \gamma$ и $\texttt{height}(P) = \ell$. Нека $0, 1, \ldots, n-1$ са всички наследници на ε в T. Понеже дървото е съвместимо с G, то $\lambda(\varepsilon) \to_G \lambda(0)\lambda(1)\cdots\lambda(n-1)$. Да положим $T_i \overset{\text{деф}}{=} i^{-1}(T)$ и $\lambda_i(\alpha) \overset{\text{деф}}{=} \lambda(i\alpha)$. Ясно е, че $P_i = (T_i, \lambda_i)$ са синтактични дървета с височина $\ell_i < \ell$. От **(И.П.)** получаваме, че $X_i \overset{\ell_i}{\lhd} \gamma_i$ като $\gamma = \gamma_0 \cdot \gamma_1 \cdots \gamma_{n-1}$. Заключаваме, че $X \overset{\ell}{\lhd} \gamma$.

□

3.2.1 Еднозначни граматики

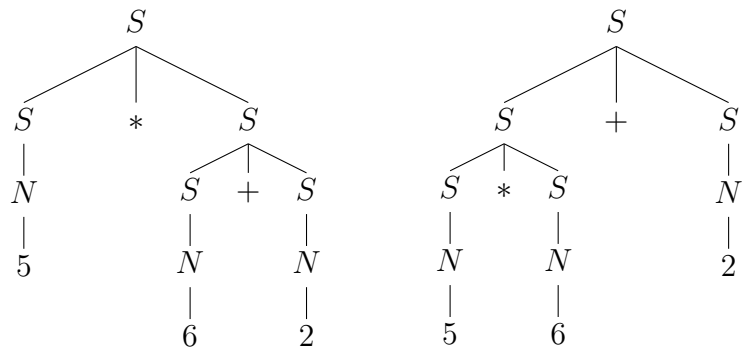
Практически, дърветата на извод са важни, когато искаме да зададем *смисъл* на една дума. Това е видно от *Пример 3.2*. Ако подходим наивно към задачата за построяване на граматика за аритметичните изрази, то смисълът на аритметичните изрази може да бъде неясен.

Пример 3.2. Да разгледаме граматика G_0 с правила

$$S \to S + S \mid S * S \mid (S) \mid N$$
$$N \to 0 \mid 1 \mid 2 \mid \ldots \mid 9$$

Подобен пример е разгледан и в [1]. Такъв вид граматики се наричат *неоднозначни* (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 3.2: Две синтактични дървета в граматиката G_0 за думата $5 * 6 + 2$.

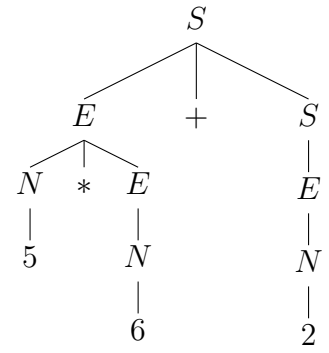
✍ Колко различни синтактични дървета за думата $1 + 5 * 6 + 2$ може да намерите?

Естествено е да искаме да модифицираме граматиката 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:

$$S \to E + S \mid E$$
$$E \to N \mid (S) \mid N * E \mid (S) * E$$
$$N \to 0 \mid 1 \mid 2 \mid \ldots \mid 9$$

$*$ има по-висок приоритет от $+$, защото $*$ се среща по-близо до листата, или аналогично, $+$ се среща по-близо до корена.

Сега думата $5 * 6 + 2$ има само едно дърво на извод. Тук операцията $*$ е с по-висок приоритет в сравнение с операцията $+$.



Фигура 3.3: Единственото синтактично дърво в граматиката G_1 за думата $5 * 6 + 2$.

Пример 3.3. Нека за езика $L = \{a^n b^m c^k \mid n + m \leq k\}$ да разгледаме граматиктите G_1 и G_2. Правилата на G_1 са

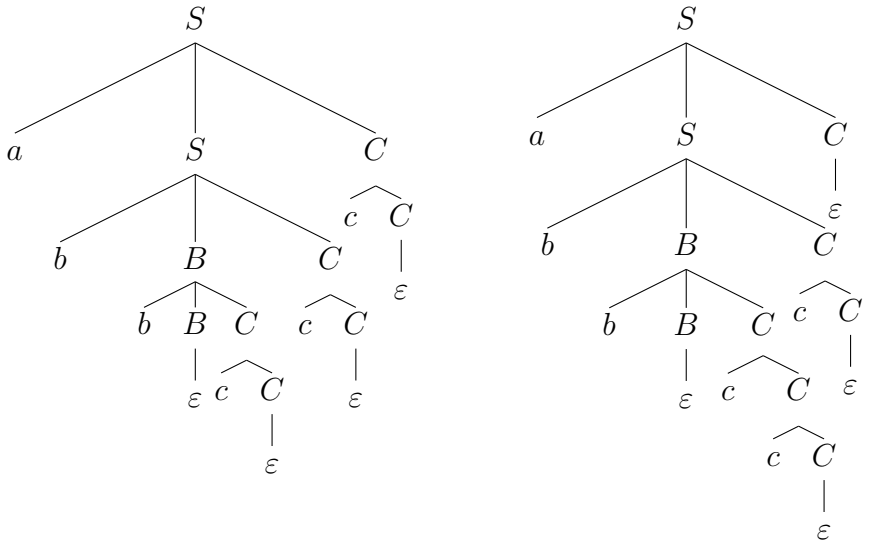
$$S \to aSC \mid bBC \mid \varepsilon$$
$$B \to bBC \mid \varepsilon$$
$$C \to cC \mid \varepsilon.$$

Правилата на G_2 са

$$S \to AC \mid \varepsilon$$
$$A \to aAc \mid bBc \mid \varepsilon$$
$$B \to bBc \mid \varepsilon$$
$$C \to cC \mid \varepsilon.$$

Лесно се съобразява, че $L = \mathcal{L}(G_1) = \mathcal{L}(G_2)$.

Да разгледаме думата $abbccc \in L$. Можем да посочим няколко синтактични дървета съвместими с G_1.



Фигура 3.4: Две синтактични дървета съвместими с G_1 за думата $abbccc$.

Посочете дума $\alpha \in L$, за която има поне две различни синтактични дървета съвместими с G_1. Съобразете, че всяка дума $\alpha \in L$ има единствено синтактично дърво съвместимо с G_2.

Ще завършим този раздел като формулираме един важен резултат, който все още нямаме сили да докажем.

Доказателството е в *Следствие 5.10*.

Теорема 3.1. Не съществува алгоритъм, който да разпознава дали дадена безконтекстна граматика е еднозначна или не.

3.2.2 Апроксимации на безконтекстен език

Обикновено използваме a, b, c за елементи на Σ и A, B, C за елементи на V. Ще използваме X за елементи на $\Sigma \cup V$.

Определение 3.3. Да напомним, че в Раздел 2.14 дефинирахме език $\mathcal{L}_{\mathcal{A}}(q)$. По подобен начин, сега ще дефинираме език $\mathcal{L}_G(X)$, за произволно $X \in \Sigma \cup V$ в граматиката G, по следния начин:

$$\mathcal{L}_G(X) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid X \stackrel{\star}{\lhd} \alpha\}.$$

Ясно е, че $\mathcal{L}(G) = \mathcal{L}_G(S)$. Нека да дефинираме и *апроксимациите* $\mathcal{L}^\ell_G(X)$ на езика $\mathcal{L}_G(X)$ по следния начин:

$$\mathcal{L}^\ell_G(X) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid X \stackrel{\leq \ell}{\lhd} \alpha\}.$$

След един бърз поглед върху дефиницията на релацията $\lhd$ веднага съобразяваме, че ако $X \in \Sigma$, то $\mathcal{L}^\ell_G(X) = \{X\}$ за всяко ℓ. Да видим сега какво става, когато $X \in V$.

Задача 3.2. 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 3.2. Нека G е безконтекстна граматика и A е променлива в G. Тогава имаме следните зависимости:

$$\mathcal{L}^0_G(A) = \emptyset$$
$$\mathcal{L}^{\ell+1}_G(A) = \bigcup\{\mathcal{L}^\ell_G(X_1) \cdots \mathcal{L}^\ell_G(X_n) \mid A \to_G X_1 \cdots X_n \text{ е правило в } G\}.$$

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 произволна дума α, за която $\alpha \in \mathcal{L}^{\ell+1}_G(A)$, т.е. $A \stackrel{\leq \ell+1}{\lhd} \alpha$. Ясно е, че е невъзможно да имаме $A \stackrel{0}{\lhd} \alpha$. Тогава, според правилата на релацията $\lhd$, имаме следния извод:

$$\frac{A \to_G X_1 \cdots X_n \qquad X_1 \overset{\leq \ell}{\lhd} \alpha_1 \qquad \cdots \qquad X_n \overset{\leq \ell}{\lhd} \alpha_n}{A \overset{\leq \ell+1}{\lhd} \underbrace{\alpha_1 \cdots \alpha_n}_{\alpha}} \text{ правило (1)}$$

Понеже $X_i \overset{\leq \ell}{\lhd} \alpha_i$, то от **(И.П.)** имаме, че $\alpha_i \in \mathcal{L}^\ell_G(X_i)$ за всяко $i = 1, 2, \ldots, n$. Тогава

$$\alpha \in \bigcup \{\mathcal{L}^\ell_G(X_1) \cdots \mathcal{L}^\ell_G(X_n) \mid A \to_G X_1 \cdots X_n\}.$$

Така доказахме включването $\subseteq$. Сега преминаваме към доказателството на включването $\supseteq$. Нека вземем произволна дума α принадлежаща на дясното множество. Това означава, че можем да представим α като $\alpha = \alpha_1 \cdots \alpha_n$, където $A \to_G X_1 \cdots X_n$ и $\alpha_i \in \mathcal{L}^\ell_G(X_i)$. Получаваме следния извод:

$$\frac{A \to_G X_1 \cdots X_n \qquad \dfrac{\alpha_1 \in \mathcal{L}^\ell_G(X_1)}{X_1 \overset{\leq \ell}{\lhd} \alpha_1} \text{ (деф.)} \qquad \cdots \qquad \dfrac{\alpha_n \in \mathcal{L}^\ell_G(X_n)}{X_n \overset{\leq \ell}{\lhd} \alpha_n} \text{ (деф.)}}{A \overset{\leq \ell+1}{\lhd} \alpha_1 \cdots \alpha_n} \text{ правило (1)}$$

Заключаваме, че думата α принадлежи на езика $\mathcal{L}^{\ell+1}(A)$. Така доказахме и включването $\supseteq$. □

Следствие 3.1. Нека G е безконтекстна граматика и A е променлива в G. Тогава

$$\mathcal{L}_G(A) = \bigcup \{\mathcal{L}_G(X_1) \cdots \mathcal{L}_G(X_n) \mid A \to_G X_1 \cdots X_n \text{ е правило в } G\}.$$

✍ Докажете сами!

Упътване. Използвайте, че $\mathcal{L}_G(A) = \bigcup_\ell \mathcal{L}^\ell_G(A)$. □

Доказателство. Нека $\alpha \in \mathcal{L}_G(A)$, т.е. $\alpha \in \mathcal{L}^{\ell+1}_G(A)$, за някое ℓ. Тогава, според *Твърдение 3.2*, $\alpha \in \mathcal{L}^\ell_G(X_1) \cdots \mathcal{L}^\ell_G(X_n)$, за някое правило $A \to_G X_1 \cdots X_n$. Оттук веднага получаваме, че $\alpha \in \mathcal{L}_G(X_1) \cdots \mathcal{L}_G(X_n)$ и следователно

$$\alpha \in \bigcup \{\mathcal{L}_G(X_1) \cdots \mathcal{L}_G(X_n) \mid A \to_G X_1 \cdots X_n \text{ е правило в } G\}.$$

За обратната посока, нека $\alpha \in \mathcal{L}_G(X_1) \cdots \mathcal{L}_G(X_n)$, където $A \to_G X_1 \cdots X_n$ е правило в G. Това означава, че $\alpha \in \mathcal{L}^\ell_G(X_1) \cdots \mathcal{L}^\ell_G(X_n)$, за някое ℓ, и следователно, според *Твърдение 3.2*, $\alpha \in \mathcal{L}^{\ell+1}_G(A)$. Заключаваме, че $\alpha \in \mathcal{L}_G(A)$. □

Да напомним, че вече видяхме, че езикът $L = \{a^n b^n \mid n \in \mathbb{N}\}$ не е регулярен.

Като първи пример, нека да видим, че съществува безконтекстен език, който не е регулярен.

Пример 3.4. 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:

$$\mathcal{L}(G) = \mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) = \{a^n b^n \mid n \in \mathbb{N}\}.$$

Примерни задачи

Задача 3.3. Докажете, че езикът $L = \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$ е безконтекстен.

Доказателство. Да разгледаме безконтекстната граматика G зададена със следните правила:

$$S \to aSc \mid B$$
$$B \to bBc \mid \varepsilon.$$

Да видим защо $\mathcal{L}(G) = \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$. Първо ще докажем *коректността* на граматиката. Това означава, че G не генерира думи извън желания език, т.е. $\mathcal{L}_G(S) \subseteq L$. 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:

$$\mathcal{L}^\ell_G(S) \subseteq \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\} \tag{3.1}$$
$$\mathcal{L}^\ell_G(B) \subseteq \{b^k c^k \mid k \in \mathbb{N}\}. \tag{3.2}$$

Това ще направим с индукция по ℓ. Да напомним, че според *Твърдение 3.2* имаме следните връзки:

$$\mathcal{L}^0_G(S) = \emptyset$$
$$\mathcal{L}^{\ell+1}_G(S) = \{a\} \cdot \mathcal{L}^\ell_G(S) \cdot \{c\} \cup \mathcal{L}^\ell_G(B)$$
$$\mathcal{L}^0_G(B) = \emptyset$$
$$\mathcal{L}^{\ell+1}_G(B) = \{b\} \cdot \mathcal{L}^\ell_G(B) \cdot \{c\} \cup \{\varepsilon\}.$$

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 $\alpha \in \mathcal{L}^{\ell+1}_G(S)$. Имаме два случая.
  - Нека $\alpha \in \{a\} \cdot \mathcal{L}^\ell_G(S) \cdot \{c\}$. 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 $\alpha \in \mathcal{L}^\ell_G(B)$. 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 $\alpha \in \mathcal{L}^{\ell+1}_G(B)$. Имаме два случая.
  - Нека $\alpha \in \{b\} \cdot \mathcal{L}^\ell_G(B) \cdot \{c\}$. 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 *пълнотата* 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 $\alpha \in \{b\} \cdot \mathcal{L}_G(B) \cdot \{c\} \subseteq \mathcal{L}_G(B)$.

Така доказахме *Свойство 3.3*. 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 можем да заключим, че $\mathcal{L}(G) = \{a^n b^k c^{n+k} \mid n, k \in \mathbb{N}\}$. □

Задача 3.4. Докажете, че езикът $L = \{a^m b^n c^k \mid m + n \geq k\}$ е безконтекстен.

Доказателство. Да разгледаме граматиката G с правила

$$S \to aSc \mid aS \mid B$$
$$B \to bBc \mid bB \mid \varepsilon.$$

От *Твърдение 3.2* имаме, че:

$$\begin{aligned}
\mathcal{L}^0_G(S) &= \emptyset \\
\mathcal{L}^{\ell+1}_G(S) &= \{a\} \cdot \mathcal{L}^{\ell}_G(S) \cdot \{c\} \cup \{a\} \cdot \mathcal{L}^{\ell}_G(S) \cup \mathcal{L}^{\ell}_G(B) \\
\mathcal{L}^0_G(B) &= \emptyset \\
\mathcal{L}^{\ell+1}_G(B) &= \{b\} \cdot \mathcal{L}^{\ell}_G(B) \cdot \{c\} \cup \{b\} \cdot \mathcal{L}^{\ell}_G(B) \cup \{\varepsilon\}.
\end{aligned}$$

Да предположим, че за произволно естествено число ℓ е изпълнено следното:

Тези две свойства ще бъдат нашето **(И.П.)** . Очевидно е, че те са изпълнени за $\ell = 0$.

$$\mathcal{L}^{\ell}_G(S) \subseteq \{a^n b^m c^k \mid n + m \geq k\} \tag{3.5}$$
$$\mathcal{L}^{\ell}_G(B) \subseteq \{b^m c^k \mid m \geq k\}. \tag{3.6}$$

Ще докажем, че:

$$\mathcal{L}^{\ell+1}_G(S) \subseteq \{a^n b^m c^k \mid n + m \geq k\}$$
$$\mathcal{L}^{\ell+1}_G(B) \subseteq \{b^m c^k \mid m \geq k\}.$$

За първото включване, да разгледаме произволна дума $\alpha \in \mathcal{L}^{\ell+1}_G(S)$. Имаме три случая:

- Ако $\alpha \in \mathcal{L}^{\ell}_G(B)$, 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 $\alpha \in \{a\} \cdot \mathcal{L}^{\ell}_G(S)$, 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 $\alpha \in \{b^m c^k \mid m \geq k\}$.

Така доказахме *коректност* на граматиката.

Най-накрая заключаваме, че
$$\mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) \subseteq \{a^n b^m c^k \mid n+m \geq k\}$$
$$\mathcal{L}_G(B) = \bigcup_\ell \mathcal{L}_G^\ell(B) \subseteq \{a^n b^m c^k \mid n+m \geq k\}.$$

Сега ще докажем *пълнота* на граматиката. Тук ще използваме представянията на $\mathcal{L}_G(S)$ и $\mathcal{L}_G(B)$ според *Следствие 3.1*.

Сега трябва да докажем обратните включвания, а именно:
$$\{a^n b^m c^k \mid n+m \geq k\} \subseteq \mathcal{L}_G(S) \tag{3.7}$$
$$\{b^m c^k \mid m \geq k\} \subseteq \mathcal{L}_G(B). \tag{3.8}$$

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 3.5. 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

$$S \to aSd \mid S_1 \mid S_2.$$

□

Задача 3.6. 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:

Ние вече знаем, че този език не е регулярен

$$\omega = \alpha[:i] \cdot \alpha[i] \cdot \alpha[i+1:] \cdot \beta[:i] \cdot \beta[i] \cdot \beta[i+1:],$$

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 3.7. 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 3.8. Докажете, че езикът $L = \{a^n b^m c^k d^\ell \mid n + k \geq m + \ell\}$ е безконтекстен.

Задача 3.9. Докажете, че езикът $L = \{\alpha\sharp\beta \mid \alpha, \beta \in \{a,b\}^\star \ \&\ |\alpha| \neq |\beta|\}$ е безконтекстен.

Задача 3.10. 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.

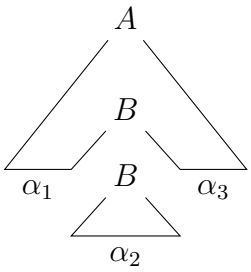
3.3 Лема за покачването

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.

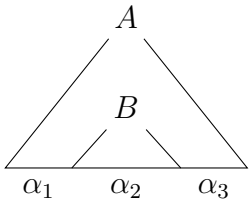
Твърдение 3.3. За произволни променливи A, B и думи $\alpha_1, \alpha_2, \alpha_3$ можем да направим извода:

$$\frac{A \overset{\ell_1}{\lhd} \alpha_1 B \alpha_3 \qquad B \overset{\ell_2}{\lhd} \alpha_2}{A \overset{\leq \ell_1 + \ell_2}{\lhd} \alpha_1 \alpha_2 \alpha_3}$$

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

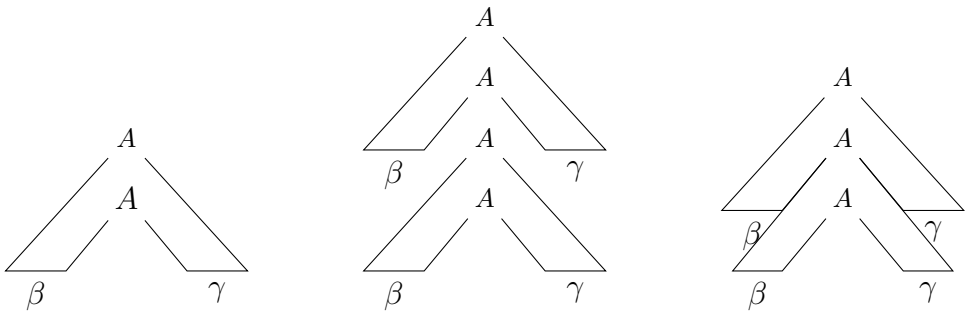
□

Твърдение 3.4. За произволна променлива A и произволни думи β и γ можем да направим извода:

$$\frac{A \overset{\ell}{\lhd} \beta A \gamma \qquad i \in \mathbb{N}}{A \overset{\leq i \cdot \ell}{\lhd} \beta^i A \gamma^i}$$

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) Синтактично дърво $P^{(2)}$ за извода $A \overset{\leq 2\ell}{\lhd} \beta^2 A \gamma^2$

□

Следващото твърдение изолзва съществено връзката между синтактичните дървета и релацията $\overset{\ell}{\lhd}$.

Твърдение 3.5. Нека G е безконтекстна граматика и

$$b \overset{\text{деф}}{=} \max\{\, |\gamma| \mid A \to \gamma \text{ е правило в } G \,\}.$$

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. Възможно е да докажем това твърдение директно, без да се позоваваме на синтактични дървета за извод, като направим индукция по ℓ. □

Числото b е максималната разклоненост, която можем да получим за синтактично дърво на извод в граматиката G.

Доказателство.

Това твърдение е на практика следствие от *Задача 1.15* в комбинация с *Твърдение 3.1*.

- Да разгледаме синтактично дърво $P = (T, \lambda)$ за извода $A \overset{\leq \ell}{\lhd} \alpha$. Според *Задача 1.15* 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 3.1 (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) $(\forall i \in \mathbb{N})[xy^iuv^iw \in L]$.

[25, стр. 125], [10, стр. 125], [9, стр. 275], [13, 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 3.5*, винаги имаме $S \overset{\ell}{\lhd} \alpha$ за $\ell \geq |V| + 1$. Нека измежду всички синтактични дървета $P = (T, \lambda)$ 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 3.7a*, 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:

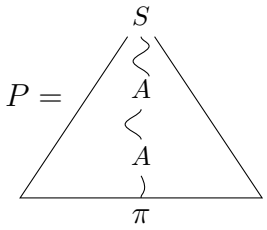
Важното е да вземем дума $\alpha \in \mathcal{L}(G)$, за която $|\alpha| > b^{|V|}$.

Принципът на Дирихле е известен също и като принципа на чекмеджетата.

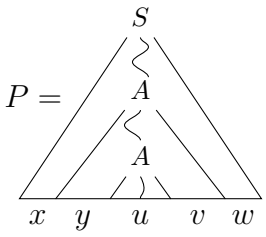
(1) $S \overset{\star}{\lhd} xAw$,

(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 .

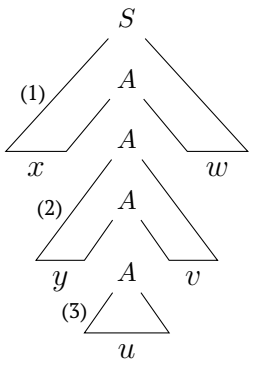
(3) $A \overset{\leq |V|}{\lhd} u$, като тук вече няма повтарящи се променливи по останалата част от пътя π.



(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:

$$\frac{S \overset{\star}{\lhd} xAw \qquad A \overset{\star}{\lhd} u}{S \overset{\star}{\lhd} \underbrace{xuw}_{\alpha},} \text{ (}\textit{Твърдение 3.3}\text{)}$$

  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 3.5* следва, че $|yuv| \leq b^{|V|+1} = p$.

- За произволно $i \in \mathbb{N}$ имаме извода:

$$\cfrac{\cfrac{\cfrac{A \overset{\star}{\lhd} yAv}{A \overset{\star}{\lhd} y^i Av^i} \quad A \overset{\star}{\lhd} u}{A \overset{\star}{\lhd} y^i uv^i} \qquad S \overset{\star}{\lhd} xAw}{S \overset{\star}{\lhd} xy^i uv^i w}$$

  (*Тв. 3.4*) (*Тв. 3.3*) (*Тв. 3.3*)

Оттук заключаваме, че $(\forall i \in \mathbb{N})[xy^i uv^i w \in \mathcal{L}(G)]$.

□

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.

[13, стр. 153].

Следствие 3.2. 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 2.4*

Следствие 3.3. 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 3.5. Да видим защо езикът $L = \{a^nb^nc^n \mid n \in \mathbb{N}\}$ не е безконтекстен.

(∀) Разглеждаме произволна константа $p \geq 1$.

(∃) Избираме дума $\alpha \in L$ със свойството, че $|\alpha| \geq p$. В случая, нека $\alpha = a^pb^pc^p$.

(∀) Разглеждаме произволно разбиване на α на пет части $\alpha = xyuvw$ със свойствата $|yuv| \leq p$ и $1 \leq |yv|$.

(∃) За всяко такова разбиване ще посочим i, за което $xy^iuv^iw \notin L$. Знаем, че поне едно от y и v не е празната дума. Имаме няколко случая за y и v.

✍ 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.

- 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 3.2* следва, че езикът L не е безконтекстен.

Твърдение 3.6. 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 3.5*, 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.
- 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

  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 3.6*.

$$L_0 = L_1 \cap L_2 = \overline{\overline{L}_1 \cup \overline{L}_2} = \overline{L}_3$$

  е безконтекстен, което е противоречие.

□

Примерни задачи

Задача 3.11. Докажете, че езикът $L = \{a^i b^j c^k \mid 0 \leq i \leq j \leq k\}$ не е безконтекстен.

Упътване. Прилагаме *Следствие 3.2*.

($\forall$) 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 3.12. Докажете, че езикът $L = \{\, \alpha \cdot \alpha \mid \alpha \in \{a,b\}^\star \,\}$ 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 3.13. 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 3.14. 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 3.15. Докажете, че езикът $L = \{\alpha\sharp\beta \mid \alpha$ е подниз на $\beta\}$ не е безконтекстен.

Упътване.

- Защо не става ако вземем $\omega = a^p \sharp a^p$?
- Защо не става ако вземем $\omega = a^p b \sharp a^p b$?
- Разгледайте $\omega = a^p b^p \sharp a^p b^p$.

□

Задача 3.16. 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) $L = \{\alpha\sharp\beta^{\mathtt{rev}} \mid \alpha, \beta \in \{0,1\}^\star \ \& \ \overline{\alpha}_{(2)} + 1 = \overline{\beta}_{(2)}\}$?

3.4 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

$$|G| \stackrel{\text{деф}}{=} |V| + |\Sigma| + \sum_{(A,\alpha)\in R} |A\alpha|.$$

3.4.1 Опростяване на безконтекстни граматики

Премахване на безполезните променливи

[10, стр. 88], [9, стр. 256].

Нека е дадена безконтекстната граматика $G = \langle V, \Sigma, R, S\rangle$. Една променлива A се нарича **полезна**, ако съществува извод от следния вид:

$$S \overset{\star}{\lhd}_G \lambda A \rho \overset{\star}{\lhd}_G \alpha,$$

където $\alpha \in \Sigma^\star$, а $\lambda, \rho \in (V \cup \Sigma)^\star$. 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 3.2. Нека е дадена безконтекстната граматика G и $\mathcal{L}(G) \neq \emptyset$. Съществува алгоритъм, който намира граматика G', за която $\mathcal{L}(G) = \mathcal{L}(G')$, със свойството, че за всяка променлива $A' \in V'$, $\mathcal{L}_{G'}(A') \neq \emptyset$.

Упътване. Целта ни е да намерим множеството `Gen` от променливи, които генерират думи, т.е. търсим множеството

$$\texttt{Gen} \overset{\text{деф}}{=} \{A \in V \mid \mathcal{L}_G(A) \neq \emptyset\}.$$

Всяка итерация на алгоритъма отнема $\mathcal{O}(|G|)$ време. Следователно, изпълнението на целия алгоритъм отнема $\mathcal{O}(|G|^2)$ време.

За целта ще построим редица от множества `Gen[n]` по следния начин:

- `Gen[0]` $\overset{\text{деф}}{=} \emptyset$;
- `Gen[n+1]` $\overset{\text{деф}}{=} \{A \in V \mid (\exists \alpha \in (\Sigma \cup$ `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 `Gen` $=$ `Gen[n]`. Това ще направим като докажем, че за всяко k, е изпълнена еквивалентността:

$$A \in \texttt{Gen[k]} \;\Leftrightarrow\; (\exists \alpha \in \Sigma^\star)[A \overset{\leq k}{\lhd}_G \alpha],$$

или с други думи,

$$A \in \texttt{Gen[k]} \;\Leftrightarrow\; \mathcal{L}^k_G(A) \neq \emptyset.$$

Дефинираме G' като $V' =$ `Gen` и правилата на G' са само тези правила на G, в които участват променливи от V' и букви от Σ. □

Следствие 3.4. Съществува *полиномиален* алгоритъм, който определя за всяка безконтекстна граматика G дали $\mathcal{L}(G) = \emptyset$.

Доказателство. Прилагаме алгоритъма от *Лема 3.2* и намираме множеството от променливи V'. Тогава $\mathcal{L}(G) = \emptyset$ точно тогава, когато $S \notin V'$. □

Лема 3.3. Съществува алгоритъм, който по дадена безконтекстна граматика $G = \langle V, \Sigma, R, S\rangle$, намира $G' = \langle V', \Sigma, R', S\rangle$, $\mathcal{L}(G') = \mathcal{L}(G)$, със свойството, че за всяко $B \in V'$ съществуват $\lambda, \rho \in (V' \cup \Sigma)^\star$, за които $S \overset{\star}{\lhd}_{G'} \lambda B \rho$, т.е. всяка променлива в G' е достижима от началната променлива S.

Упътване. Нашата цел е да намерим множеството

$$\texttt{Reach} \overset{\text{деф}}{=} \{B \in V \mid S \overset{\star}{\lhd} \lambda B \rho, \text{ за някои } \lambda, \rho \in (\Sigma \cup V)^\star\}.$$

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 `Reach`. За да постигнем тази цел, ще започнем да строим множествата $\texttt{Reach[i]} \subseteq V$ по следния начин:

$$\begin{aligned}
\texttt{Reach[0]} &\overset{\text{деф}}{=} \{S\} \\
\texttt{Reach[i+1]} &\overset{\text{деф}}{=} \{B \in V \mid (\exists A \in \texttt{Reach[i]})(\exists \lambda, \rho \in (V \cup \Sigma)^\star)[\, A \to_G \lambda B \rho \,]\} \\
&\quad \cup \texttt{Reach[i]}.
\end{aligned}$$

Докажете, че за всяко i е изпълнено:

$$\texttt{Reach[i]} = \{B \in V \mid S \overset{\leq i}{\lhd} \lambda B \rho, \text{ за някои } \lambda, \rho \in (\Sigma \cup V)^\star\}.$$

Спираме да строим тези множества, когато достигнем до стъпка n, за която

$$\texttt{Reach[n]} = \texttt{Reach[n+1]}.$$

✍ Защо сме сигурни, че ще достигнем такава стъпка?

Съобразете, че за това n е изпълнено, че $\texttt{Reach[n]} = \texttt{Reach}$, т.е.

$$\texttt{Reach[n]} = \{B \in V \mid S \overset{\star}{\lhd} \lambda B \rho, \text{ за някои } \lambda, \rho \in (\Sigma \cup V)^\star\}.$$

□

Пример 3.6. 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.

- `Gen[0]` $= \emptyset$;
- `Gen[1]` $= \{A, C\}$, защото $A \to a$ и $C \to \varepsilon$;
- `Gen[2]` $= \{A, C, S\}$, защото $S \to aA$;
- не можем да добавим B към `Gen[3]`, следователно `Gen[3]` $=$ `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.

- `Reach[0]` $= \{S\}$;
- `Reach[1]` $= \{S, A\}$, защото $S \to aAa$;
- `Reach[2]` $= \{S, A\}$.

Така получаваме граматиката G'' със следните правила:

$$S \to aA$$
$$A \to a \mid aAa.$$

Теорема 3.2. 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.

✍ 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 3.3*, то нищо няма да премахнем, защото A и B са достижими от S. След това, ако приложим процедурата от *Лема 3.2*, то ще премахнем всички правила, в които участва B. Така A става недостижима от S и трябва пак да приложим процедурата от *Лема 3.3*. Алгоритъмът описан тук е квадратичен. Има и линеен такъв. Вижте [9, стр. 296]

Упътване. Прилагаме върху G първо процедурата от *Лема 3.2* и след това върху резултата прилагаме процедурата от *Лема 3.3*. □

Задача 3.17. 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 3.4. 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:

На всяка стъпка трябва да обходим цялата граматика, следователно всяка итерация отнема $\mathcal{O}(|G|)$ време.

Обърнете внимание, че имаме `E[n]`*, а не просто `E[n]`. Също така да напомним, че $\emptyset^\star = \{\varepsilon\}$.

   - `E[0]` $\overset{\text{деф}}{=} \emptyset$;
   - `E[n+1]` $\overset{\text{деф}}{=} \{B \in V \mid (\exists \alpha \in \texttt{E[n]}^\star)[B \to_G \alpha]\}$.
   - Докажете, че за всяко n е изпълнено, че

$$\texttt{E[n]} = \{A \in V \mid A \overset{\leq n}{\lhd} \varepsilon\}.$$

   - Спираме на първото n, за което `E[n]` $=$ `E[n+1]`. Лесно се съобразява, че за това n е изпълнено, че

Намираме множеството `E` за време $\mathcal{O}(|G|^2)$.

$$\texttt{E[n]} = \{A \in V \mid A \overset{\star}{\lhd} \varepsilon\}.$$

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, за $i = 1, \ldots, k$, са ε. В противен случай, така бихме добавили правило $A \to_G \varepsilon$.

   Лесно се съобразява, че $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

✍ Докажете сами!

□

Пример 3.7. 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 можем да намерим множеството от променливи E, от които се извежда ε в граматиката G като започнем да намираме неговите апроксимации:

- `E[0]` $\overset{\text{деф}}{=} \emptyset$;
- `E[1]` $= \{C, U\}$, защото $C \to \varepsilon$ и $U \to \varepsilon$;
- `E[2]` $= \{C, U, B\}$, защото $B \to UC$;
- `E[3]` $= \{C, U, B, A\}$, защото $A \to BC$;
- `E[4]` $=$ `E[3]`.
- Оттук е ясно, че $(\forall n \geq 3)[\texttt{E[n]} = \texttt{E[3]}]$. Заключаваме, че

$$\texttt{E} = \{X \in V \mid X \overset{\star}{\lhd} \varepsilon\} = \{C, U, B, A\}.$$

Това означава, че $\varepsilon \notin \mathcal{L}(G)$, защото $S \notin$ E. 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 [9, стр. 263] преименуващите правила се наричат *unit productions*.

Едно правило в граматиката G се нарича **преименуващо**, ако е от вида $A \to_G B$. 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 $A \overset{\star}{\lhd} B$, ако $B \to \alpha$ е правило в граматиката G, което не е преименуващо, то добавяме към G' правилото $A \to \alpha$.

Лема 3.5. 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 $\mathcal{L}(G') = \mathcal{L}(G)$.

Упътване. Първо намираме множеството от двойки променливи от следния вид:

$$\texttt{Unit} = \{(A, B) \in V \times V \mid A \overset{\star}{\lhd} B\}.$$

Отново, за да намерим `Unit`, ние строим неговите апроксимации `Unit[n]`, където:

$$
\begin{aligned}
&\texttt{Unit[0]} \overset{\text{деф}}{=} \{(A, A) \mid A \in V\}\\
&\texttt{Unit[n+1]} \overset{\text{деф}}{=} \texttt{Unit[n]} \cup \{(A, B) \mid (\exists C)[A \to_G C \;\&\; (C, B) \in \texttt{Unit[n]}]\}.
\end{aligned}
$$

Лесно се съобразява, че имаме свойството:

$$\texttt{Unit[n]} = \{(A, B) \in V \times V \mid A \overset{\leq n}{\lhd} B\}.$$

`Unit` има големина $\mathcal{O}(|G|^2)$.

Спираме на първото n, за което `Unit[n]` $=$ `Unit[n+1]`. Тогава `Unit` $=$ `Unit[n]`.

Нека $R'_0 \overset{\text{деф}}{=} R \setminus (V \times V)$ са правилата на R, които не са преименуващи. Тогава правилата на новата граматика G' ще бъдат:

$$R' \overset{\text{деф}}{=} \{\ (A, \alpha) \in V \times (V \cup \Sigma)^\star \mid (\exists B)[(A, B) \in \texttt{Unit} \;\&\; (B, \alpha) \in R'_0]\ \}.$$

Обърнете внимание, че понеже $(A, A) \in \texttt{Unit}$ за всяко $A \in V$, то $R'_0 \subseteq R'$. Лесно се съобразява, че $\mathcal{L}(G) = \mathcal{L}(G')$. □

Пример 3.8. 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 `Unit`.

$$\texttt{Unit[0]} = \{(S,S),(A,A),(B,B),(C,C)\};$$
$$\texttt{Unit[1]} = \{(S,S),(S,B),(A,A),(A,S),(B,B),(B,C),(C,C)\};$$
$$\texttt{Unit[2]} = \{(S,S),(S,B),(S,C),(B,B),(B,C),(C,C),(A,A),(A,S),(A,B)\};$$
$$\texttt{Unit[3]} = \{(S,S),(S,B),(S,C),(B,B),(B,C),(C,C),(A,A),(A,S),(A,B),(A,C)\};$$
$$\texttt{Unit[4]} = \{(S,S),(S,B),(S,C),(B,B),(B,C),(C,C),(A,A),(A,S),(A,B),(A,C)\}.$$

Получихме, че $\texttt{Unit[3]} = \texttt{Unit[4]}$. Оттук можем да заключим следното:

$$A \overset{\star}{\lhd} A, B, S, C$$
$$B \overset{\star}{\lhd} B, C$$
$$S \overset{\star}{\lhd} S, B, C$$
$$C \overset{\star}{\lhd} C.$$

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 $A \overset{\star}{\lhd} B, S, C$, то добавяме към R' правилата:

$$A \to BC \mid AB \mid a \mid b \mid CC.$$

- Понеже имаме,че $B \overset{\star}{\lhd}_G C$, то добавяме към R' правилата:

$$B \to AB \mid a \mid b.$$

- Понеже имаме, че $S \overset{\star}{\lhd}_G B, C$, то добавяме към R' правилата:

$$S \to BC \mid AB \mid a \mid b.$$

Накрая получаваме, че граматиката G' има правила

$$S \to BC \mid AB \mid CC \mid a \mid b$$
$$A \to BS \mid BC \mid AB \mid a \mid b \mid CC$$
$$B \to AB \mid a \mid b \mid BC$$
$$C \to AB \mid a \mid b.$$

Задача 3.18. Премахнете преименуващите правила от граматиката G, като запазите езика, ако G има следните правила:

$$S \to C \mid CC \mid b$$
$$A \to B$$
$$B \to S \mid C \mid BC$$
$$C \to a \mid AB;$$

Премахване на дългите правила

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,

Новата граматика ще има дължина $\mathcal{O}(|G|)$.

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 3.19. 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:

$$\begin{aligned} S &\to CC \mid b \\ A &\to BSB \mid a \\ B &\to ba \mid BC \\ C &\to BaSA \mid a \mid b. \end{aligned}$$

3.4.2 Нормална Форма на Чомски

Една безконтекстна граматика е в **нормална форма на Чомски** (НФЧ), ако всяко правило е от вида

$$A \to BC \text{ и } A \to a,$$

където A, B, C са произволни променливи и a е произволна буква.

Теорема 3.3. 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 $\mathcal{O}(|G|)$.
- 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 $S \to \alpha$.

Редът на стъпките е важен. Трябва преди това сме премахнали дългите правила. Вижте [9, стр. 296].

□

Теорема 3.4. 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 $\mathcal{O}(|G|^2)$.

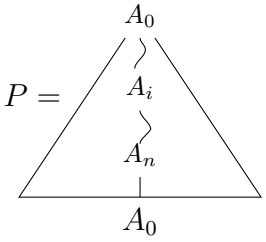
Теорема 3.5. Съществува *полиномиален* алгоритъм, който определя по дадена безконтекстна граматика G дали $\mathcal{L}(G)$ е безкраен език.

[10, стр. 137]. Важно е, че алгоритмът е полиномиален. ✍ Защо от *Лема 3.1* имаме експоненциален алгоритъм за проверка дали езикът на една граматика е безкраен?

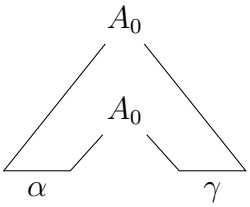
Доказателство. Нека е дадена една безконтекстна граматика G. Нека да разгледаме граматиката G' в НФЧ *без безполезни променливи*, за която $\mathcal{L}(G) = \mathcal{L}(G')$. От граматиката $G' = \langle V', \Sigma, S, R' \rangle$ строим граф с възли променливите от V' като за $A, B \in V'$ имаме ребро $A \to B$ 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 $\lambda \overset{\star}{\lhd} \alpha$, където $\alpha \in \Sigma^\star$ и $|\lambda| \leq |\alpha|$ и $\rho \overset{\star}{\lhd} \gamma$, 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 3.3* и *Твърдение 3.4* 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 да 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. □

Забележка. Ясно е, че ако изискваме една граматика да бъде в НФЧ, то $\varepsilon \notin \mathcal{L}(G)$. Ако все пак искаме ε да бъде в езика на граматиката G, то можем

да позволим от началната променлива S да имаме правилото $S \to \varepsilon$, но тогава забраняваме S да се среща в дясна страна на правило.

За да направим това, по дадена граматика G в НФЧ, трябва да добавим нова начална променлива S' и да добавим правилата $S' \to \alpha$, където $S \to \alpha$ е правило в граматиката G. Така ще получим нова граматика G', за която $\mathcal{L}(G') = \mathcal{L}(G) \cup \{\varepsilon\}$.

Да напомним, че всички от изброените тук проблеми са алгоритмично разрешими за регулярни езици.

Проблемът за празнота: Да. *Следствие 3.4* ни казва, че съществува алгоритъм, който разрешава проблема за празнота за безконтекстни езици.

Проблемът за безкрайност: Да. От *Теорема 3.5* знаем, че този проблем е алгоритмично разрешим.

Проблемът за универсалност: Не. Чак в *Теорема 5.9* ще видим, че не съществува алгоритъм, който да разрешава този проблем за безконтекстни езици.

Проблемът за принадлежност: Да. Според *Теорема 3.6* проблемът за принадлежност е алгоритмично разрешим за безконтекстни езици.

Проблемът за включване: Не. Да фиксираме безконтекстна граматика G_0, за която $\mathcal{L}(G_0) = \Sigma^\star$. Съобразете, че за произволна безконтекстна граматика G имаме еквивалентността:

$$\mathcal{L}(G) = \Sigma^\star \;\Leftrightarrow\; \mathcal{L}(G_0) \subseteq \mathcal{L}(G).$$

Това означава, че ако допуснем, че проблемът за включване на безконтекстни граматики е алгоритмично разрешим, то проблемът за универсалност за безконтекстни граматики също би бил разрешим. Но това е противоречие. Заключаваме, че проблемът за включване не е разрешим.

Проблемът за равенство: Не. Аналогично се съобразява, че този проблем също не е алгортимично разрешим.

3.4.3 Проблемът за принадлежност

Ние знаем, че съществува доста прост алгоритъм, който разрешава проблема за принадлежност за автоматни езици. Сега ще видим, че този проблем е алгоритмично разрешим и за безконтекстни езици.

Теорема 3.6. 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 [13, стр. 192], [24, стр. 141]. При вход дума α, алгоритъмът работи за време $\mathcal{O}(|\alpha|^3)$.

Algorithm 6 Проблемът за принадлежност за безконтекстни езици

```
1:  procedure BELONG(G, α)
2:      n := len(α)                                  ▷ Вход дума α = a_0 a_1 ··· a_{n-1}
3:      for all i < n do
4:          V[i][i] := {A ∈ V | A →_G α[i]}
5:          for all j := i + 1, ..., n − 1 do
6:              V[i][j] := ∅
7:      for all 1 ≤ s < n do                         ▷ Дължина на интервала
8:          for all i < n − s do                     ▷ Начало на интервала
9:              for all i ≤ k < i + s do             ▷ Разделяне на интервала
10:                 if ∃(A, BC) ∈ R & B ∈ V[i][k] & C ∈ V[k+1][i+s] then
11:                     V[i][i+s] := V[i][i+s] ∪ {A}
12:     if S ∈ V[0][n-1] then
13:         return True                              ▷ Има извод на думата от S
14:     else
15:         return False
```

Лема 3.6. Нека е дадена безконтекстната граматика G в нормална форма на Чомски и думата α. Всеки път, точно преди да се изпълни ред (7) от програмата описана в Алгоритъм 6, е изпълнено, че за всяко $i < n - s$,

$$\texttt{V[i][i+s]} = \{A \in V \mid A \overset{\star}{\lhd}_G a_i a_{i+1} \cdots a_{i+s}\}.$$

Доказателство. Пълна индукция по s. За $s = 0$ е ясно, защото от ред (4) и от факта, че граматиката е в нормална форма на Чомски следва, че за всяко $i < n$,

$$\texttt{V[i][i]} = \{A \in V \mid A \to_G \alpha[i]\} = \{A \in V \mid A \overset{\star}{\lhd}_G a_i\}.$$

Сега ще докажем твърдението за $s > 0$, т.е. ще докажем, че за всяко $i < n - s$ е изпълнено, че:

$$\texttt{V[i][i+s]} = \{A \in V \mid A \overset{\star}{\lhd}_G a_i a_{i+1} \cdots a_{i+s}\}.$$

Да разгледаме двете посоки на това равенство.

(⊆) 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 (11). Тогава имаме, че съществува k, за което $i \leq k < i + s$, и са изпълнени условията:

- $B \in$ `V[i][k]`, като $k = i + \overbrace{(k - i)}^{s'}$;
- $C \in$ `V[k+1][i+s]`, като $i + s = (k + 1) + \overbrace{(i + s - k - 1)}^{s''}$;
- $A \to BC$ е правило в граматиката G.

Понеже $s' < s$ и $s'' < s$, от индукционното предположение имаме следното:

$$B \in \texttt{V[i][k]} \overset{\textbf{(И.П.)}}{\Longrightarrow} B \overset{\star}{\lhd}_G a_i a_{i+1} \cdots a_k$$
$$C \in \texttt{V[k+1][i+2]} \overset{\textbf{(И.П.)}}{\Longrightarrow} C \overset{\star}{\lhd}_G a_{k+1} \cdots a_{i+s}.$$

Заключаваме веднага, че $A \overset{\star}{\lhd}_G a_i a_{i+1} \cdots a_{i+s}$.

(⊇) Нека $A \overset{\star}{\lhd} a_i a_{i+1} \cdots a_{i+s}$ 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 съществува k, за което $i \leq k < i + s$, и

$$B \overset{\star}{\lhd} a_i \cdots a_k$$
$$C \overset{\star}{\lhd} a_{k+1} \cdots a_{i+s}.$$

От **(И.П.)** получаваме, че $B \in$ `V[i][k]` и $C \in$ `V[k+1][i+s]`. Тогава от ред (11) на алгоритъма е ясно, че $A \in$ `V[i][i+s]`.

□

Пример 3.9. Нека е дадена граматиката G в нормална форма на Чомски с правила

$$S \to a \mid AB \mid AC$$
$$A \to a$$
$$B \to b$$
$$C \to SB \mid AS.$$

Ще приложим CYK алгоритъма за да проверим дали $aaabb \in \mathcal{L}(G)$.

- За $s = 0$ имаме, че:
  - `V[0][0]` $=$ `V[1][1]` $=$ `V[2][2]` $= \{S, A\}$;
  - `V[3][3]` $=$ `V[4][4]` $= \{B\}$.
- За $s = 1$ имаме, че:
  - `V[0][1]` $=$ `V[1][2]` $= \{C\}$;
  - `V[2][3]` $= \{S, C\}$;
  - `V[3][4]` $= \emptyset$.
- За $s = 2$ имаме, че:
  - `V[0][2]` $= \{S\} \cup \emptyset$;
  - `V[1][3]` $= \{S, C\} \cup \emptyset$;
  - `V[2][4]` $= \emptyset \cup \{C\}$.
- За $s = 3$ имаме, че:

  - `V[0][3]` $= \{S, C\} \cup \emptyset \cup \emptyset = \{S, C\}$;
  - `V[1][4]` $= \{S\} \cup \emptyset \cup \{C\} = \{S, C\}$.
- За $s = 4$ имаме, че:
  - `V[0][4]` $= \{S, C\} \cup \emptyset \cup \emptyset \cup \{C\} = \{S, C\}$.

Понеже $S \in$ `V[0][4]`, то $aaabb \in \mathcal{L}(G)$.

3.5 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.

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.

Означаваме

$$\Sigma_\varepsilon \stackrel{\text{деф}}{=} \Sigma \cup \{\varepsilon\}$$
$$\Gamma^{\leq 2} \stackrel{\text{деф}}{=} \{\varepsilon\} \cup \Gamma \cup \Gamma^2.$$

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 [7, стр. 131]. На англ. моментно описание - instanteneous description.

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.

$$\frac{\Delta(q, x, A) \ni (p, \beta) \qquad x \in \Sigma_\varepsilon \qquad \gamma \in \Gamma^\star}{(q, x\alpha, A\gamma) \vdash_P (p, \alpha, \beta\gamma)}$$

Фигура 3.9: Правило 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:

$$\frac{}{\kappa \vdash_P^0 \kappa} \text{ (рефлексивност)} \qquad \frac{\kappa \vdash_P \kappa'' \qquad \kappa'' \vdash_P^\ell \kappa'}{\kappa \vdash_P^{\ell+1} \kappa'} \text{ (транзитивност)}$$

Фигура 3.10: Дефиниция на релацията $\vdash_P^\ell$

Задача 3.20. Докажете, че:

$$\frac{\kappa \vdash_P^{\ell_1} \kappa_1 \qquad \kappa_1 \vdash_P^{\ell_2} \kappa_2}{\kappa \vdash_P^{\ell_1+\ell_2} \kappa_2}$$

Упътване. Индукция по ℓ_1. □

Ако не се интересуваме от точния брой стъпки в едно изчисление, ще използваме релацията $\vdash_P^\star$, която е дефинирата по следния начин:

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

С други думи, бинарната релация $\vdash_P^\star$ е рефлексивното и транзитивното затваряне на бинарната релация $\vdash_P$.

Определение 3.4. Езикът $\mathcal{L}(P)$, който се разпознава от стековия автомат P дефинираме по следния начин:

$$\mathcal{L}(P) \overset{\text{деф}}{=} \{\, \omega \in \Sigma^\star \mid (q_{\texttt{start}}, \omega, \sharp) \vdash_P^\star (q_{\texttt{accept}}, \varepsilon, \varepsilon) \,\}.$$

Възможно е да се дадат и други еквивалентни дефиниции на $\mathcal{L}(P)$.

Сега ще разгледаме две важни свойства на релацията $\vdash_P^\ell$, които ще ни бъдат полезни по-нататък. Първото свойство ни казва как можем да „слепваме" две изчисления в едно голямо изчисление.

Задача 3.21. Докажете, че:

$$\frac{(p, \alpha_1, C_1) \vdash_P^{\ell_1} (r, \varepsilon, \varepsilon) \qquad (r, \alpha_2, C_2) \vdash_P^{\ell_2} (q, \varepsilon, \varepsilon)}{(p, \alpha_1\alpha_2, C_1C_2) \vdash_P^{\ell_1+\ell_2} (q, \varepsilon, \varepsilon)}$$

Упътване. Достатъчно е да проследим извода:

$$\frac{\dfrac{(p, \alpha_1, C_1) \vdash_P^{\ell_1} (r, \varepsilon, \varepsilon)}{(p, \alpha_1\alpha_2, C_1C_2) \vdash_P^{\ell_1} (r, \alpha_2, C_2)} \qquad (r, \alpha_2, C_2) \vdash_P^{\ell_2} (q, \varepsilon, \varepsilon)}{(p, \alpha_1\alpha_2, C_1C_2) \vdash_P^{\ell_1+\ell_2} (q, \varepsilon, \varepsilon)}$$

□

Второто свойство ни казва кога можем да разбием едно изчисление на по-малки части. Това свойство ще бъде важно по-нататък.

Твърдение 3.7. Нека $(q, \alpha, A\rho) \vdash_P^{\ell} (p, \varepsilon, \varepsilon)$. Тогава съществуват ℓ_1, ℓ_2, за които $\ell_1 + \ell_2 = \ell$, състояние r и разбиване на α като $\alpha = \alpha_1\alpha_2$, така че:

- $(q, \alpha_1, A) \vdash_P^{\ell_1} (r, \varepsilon, \varepsilon)$;
- $(r, \alpha_2, \rho) \vdash_P^{\ell_2} (p, \varepsilon, \varepsilon)$.

Упътване. Пълна индукция по ℓ. Ясно е, че трябва $\ell \geq 1$. За $\ell = 1$, то е ясно, че трябва $\rho = \varepsilon$ и единствената стъпка в изчислението е направена заради $\Delta(q, x, A) \ni (r, \varepsilon)$. Тогава е ясно, че $\alpha = x$ и $r = p$. Получаваме, че в този случай изчислението се разбива не две части по тривиален начин:

- $(q, \underbrace{x}_{\alpha_1}, A) \vdash_P^1 (r, \varepsilon, \varepsilon)$;
- $(r, \underbrace{\varepsilon}_{\alpha_2}, \varepsilon) \vdash_P^0 (r, \varepsilon, \varepsilon)$.

Нека $\ell > 1$. Трябва да разгледаме няколко случая.

- Ако първата стъпка в изчислението е направена заради $\Delta(q, x, A) \ni (q', BC)$, където $x \in \Sigma_\varepsilon$, то $(q', \alpha', BC\rho) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)$, където $\alpha = x\alpha'$. Тогава от **(И.П.)** получаваме следното:

  (1) $(q', \beta_1, B) \vdash_P^{\ell'_1} (r', \varepsilon, \varepsilon)$;

  (2) $(r', \beta_2, C\rho) \vdash_P^{\ell'_2} (p, \varepsilon, \varepsilon)$,

  за някое състояние r', където $\ell'_1 + \ell'_2 = \ell - 1$ и $\beta_1\beta_2 = \alpha'$. Понеже $\ell'_2 < \ell$, отново от **(И.П.)** получаваме, че изчислението $(r', \beta_2, C\rho) \vdash_P^{\ell'_2} (p, \varepsilon, \varepsilon)$ може да се разбие така:

  (3) $(r', \gamma_1, C) \vdash_P^{k_1} (r'', \varepsilon, \varepsilon)$;

  (4) $(r'', \gamma_2, \rho) \vdash_P^{k_2} (p, \varepsilon, \varepsilon)$,

  за някое състояние r'', където $k_1 + k_2 = \ell'_2$ и $\gamma_1\gamma_2 = \beta_2$. Можем да комбинираме първата стъпка от изчислението с частичните изчисления (1) и (3) и да заключим следното:

  – $(q, \underbrace{x\beta_1}_{\alpha_1}, A) \vdash_P^{1+\ell'_1+k_1} (r'', \varepsilon, \varepsilon)$;

  – $(r'', \underbrace{\gamma_2}_{\alpha_2}, \rho) \vdash_P^{k_2} (p, \varepsilon, \varepsilon)$.

- Вече свършихме тежката работа. Нека все пак за пълнота да разгледаме и другите случаи. Ако първата стъпка в изчислението е направена заради $\Delta(q, x, A) \ni (q', B)$, то $(q', \alpha', B\rho) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)$, където $\alpha = x\alpha'$. Тук ведната от **(И.П.)** получаваме, че можем да разбием изчислението така:
  - $(q', \alpha'_1, B) \vdash_P^{\ell'_1} (r', \varepsilon, \varepsilon)$;
  - $(r', \alpha'_2, \rho) \vdash_P^{\ell'_2} (p, \varepsilon, \varepsilon)$,

  за някое състояние r', където $\alpha'_1\alpha'_2 = \alpha'$. Оттук заключаваме, че:
  - $(q, \underbrace{x\alpha'_1}_{\alpha_1}, A) \vdash_P^{1+\ell'_1} (r', \vdash, \varepsilon)$;
  - $(r', \underbrace{\alpha'_2}_{\alpha_2}, \rho) \vdash_P^{\ell'_2} (p, \varepsilon, \varepsilon)$.
- Ако първата стъпка в изчислението е направена заради $\Delta(q, x, A) \ni (q', \varepsilon)$, то нека $\alpha = x\alpha'$. Тогава всичко е ясно, защото можем да разбием изчислението така:
  - $(q, \underbrace{x}_{\alpha_1}, A) \vdash_P^1 (q', \varepsilon, \varepsilon)$;
  - $(q', \underbrace{\alpha'}_{\alpha_2}, \rho) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)$.

□

Примери

Пример 3.10. За езика $L = \{a^n b^n \mid n \in \mathbb{N}\}$, да разгледаме стековия автомат

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle,$$

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:

  (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 3.11. Езикът $L = \{\, \omega\omega^{\texttt{rev}} \mid \omega \in \{a, b\}^\star \,\}$ се разпознава от стеков автомат

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle,$$

където:

- $Q \stackrel{\text{деф}}{=} \{q, p, f\}$ и $q_{\texttt{start}} \stackrel{\text{деф}}{=} q$, $q_{\texttt{accept}} \stackrel{\text{деф}}{=} f$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$, $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$;
- 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 $\omega\omega^{\texttt{rev}}$ може да се запише като $\omega_1 aa\omega_1^{\texttt{rev}}$ или $\omega_1 bb\omega_1^{\texttt{rev}}$. 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 $\omega^{\texttt{rev}}$. 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:

$$|\alpha| = n \implies (q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \alpha^{\texttt{rev}}\sharp) \tag{3.13}$$
$$|\beta| = n \implies (p, \beta, \beta\sharp) \vdash_P^n (p, \varepsilon, \sharp). \tag{3.14}$$

Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете с индукция по n, че за всяко естествено число n са изпълнени свойствата:

$$(q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \gamma\sharp) \implies \gamma = \alpha^{\texttt{rev}} \,\&\, |\alpha| = n \tag{3.15}$$
$$(p, \beta, \gamma\sharp) \vdash_P^n (p, \varepsilon, \sharp) \implies \gamma = \beta \,\&\, |\beta| = n. \tag{3.16}$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

Пример 3.12. Езикът $L = \{\, \omega \in \{a,b\}^\star \mid |\omega|_a = |\omega|_b \}$ се разпознава от стековия автомат

От *Задача 3.24* знаем, че този език е безконтекстен.

$$P = \langle Q, \Sigma, \Gamma, \sharp, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle,$$

където:

- $Q = \{q, f\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, b, \sharp\}$;
- $q_{\texttt{start}} = q$ и $q_{\texttt{accept}} = f$;
- 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:

$$\begin{aligned}
(q, \overbrace{a^k b\gamma'}^{\gamma}, a^n\sharp) &\vdash_P^\star (q, b\gamma', a^{n+k}\sharp) && \text{// правило (3)}\\
&\vdash_P^\star (q, \gamma', a^{n+k-1}\sharp) && \text{// правило (5)}\\
&\vdash_P^\star (q, \varepsilon, \sharp). && \text{// \textbf{(И.П.)}}
\end{aligned}$$

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:

$$(\forall n)[(q,\gamma,a^n\sharp) \vdash_P^\star (q,\varepsilon,\sharp) \implies a^n\gamma \in L] \tag{3.19}$$
$$(\forall n)[(q,\gamma,b^n\sharp) \vdash_P^\star (q,\varepsilon,\sharp) \implies b^n\gamma \in L]. \tag{3.20}$$

Нека имаме изчислението $(q,\gamma,a^n\sharp) \vdash_P^\star (q,\varepsilon,\sharp)$. 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:

$$\begin{aligned}(q,a^kb\gamma',a^n\sharp) &\vdash_P^\star (q,b\gamma',a^{n+k}\sharp)\\ &\vdash_P (q,\gamma',a^{n+k-1}\sharp)\\ &\vdash^\star (q,\varepsilon,\sharp).\end{aligned}$$

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

От *Задача 3.25* знаем, че този език е безконтекстен.

Пример 3.13. 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:

- $Q = \{q,f\}$;
- $q_{\texttt{start}} = q$ и $q_{\texttt{accept}} = f$;
- $\Sigma = \{a,b\}$ и $\Gamma = \{a,\sharp\}$;

✍ Докажете, че $L = \mathcal{L}(P)$!

- Можем да дефинираме релацията на преходите Δ по следния начин:
  (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 $\mathcal{L}(P) \subseteq L$.

3.6 Теорема за еквивалентност

Лема 3.7. За всяка безконтекстна граматика G, съществува стеков автомат P, такъв че $\mathcal{L}(G) = \mathcal{L}(P)$.

Доказателството на лемата следва до голяма степен [18, 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 $\mathcal{L}(G)$.

- $Q = \{q_{\texttt{start}}, p, q_{\texttt{accept}}\}$;
- $\Gamma = \Sigma \cup V \cup \{\sharp\}$;
- Функцията на преходите Δ за стековия автомат P дефинираме по следния начин:

Понеже граматиката е в нормална форма на Чомски, то $|\alpha| \leq 2$ и удовлетворяваме дефиницията на Δ.

  (1) $\Delta(q_{\texttt{start}}, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(p, S\sharp)\}$;
  (2) За всяка променлива $A \in V$,

$$\Delta(p, \varepsilon, A) \stackrel{\text{деф}}{=} \{(p, \alpha) \mid A \to_G \alpha \text{ е правило в } G\}.$$

  (3) За всяка буква $a \in \Sigma$,

$$\Delta(p, a, a) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}.$$

  (4) $\Delta(p, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(q_{\texttt{accept}}, \varepsilon)\}$.

Ще докажем, че за всяка променлива $A \in V$ и всяка дума $\alpha \in \Sigma^\star$, е изпълнено, че:

(а) ако $A \overset{\star}{\lhd} \alpha$, то $(p, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon)$;

(б) ако $(p, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon)$, то $A \overset{\star}{\lhd} \alpha$.

Ако приемем, че (а) и (б) са изпълнени, тогава, ако вземем $A = S$, то ще получим следната верига от еквивалентности:

$$\begin{aligned}
\alpha \in \mathcal{L}(G) &\Leftrightarrow S \overset{\star}{\lhd} \alpha \\
&\Leftrightarrow (p, \alpha, S) \vdash_P^\star (p, \varepsilon, \varepsilon) && \text{// от (а) и (б)} \\
&\Leftrightarrow (q_{\texttt{start}}, \alpha, \sharp) \vdash_P^\star (q_{\texttt{accept}}, \varepsilon, \varepsilon) && \text{// от деф. на } P \\
&\Leftrightarrow \alpha \in \mathcal{L}(P).
\end{aligned}$$

Сега преминаваме към доказателствата на двете твърдения.

Доказателството на (а) ще проведем с *пълна индукция* по дължината ℓ на извода $A \overset{\ell}{\lhd} \alpha$. За целта, да разгледаме свойството

$$R(\ell) \overset{\text{деф}}{=} (\forall A \in V)(\forall \alpha \in \Sigma^\star)[\, A \overset{\ell}{\lhd} \alpha \implies (p, \alpha, A) \vdash^\star (p, \varepsilon, \varepsilon)\,].$$

Ще докажем, че $(\forall \ell \in \mathbb{N}) R(\ell)$.

Нека приемем, че $A \overset{\ell}{\lhd} \alpha$. За да докажем, че $(p, \alpha, A) \vdash^\star (p, \varepsilon, \varepsilon)$, да видим как този извод е получен. Първо, нека имаме следния извод:

$$\frac{A \to_G a}{A \overset{1}{\lhd} \underbrace{a}_{\alpha}}$$

Тогава според конструкцията на стековия автомат P, имаме изчислението:

$$\dfrac{\dfrac{\dfrac{A \to_G a}{\Delta(p, \varepsilon, A) \ni (p, a)}}{(p, a, A) \vdash_P (p, a, a)} \quad \dfrac{\Delta(p, a, a) \ni (p, \varepsilon)}{(p, a, a) \vdash_P (p, \varepsilon, \varepsilon)}}{(p, a, A) \vdash_P^2 (p, \varepsilon, \varepsilon)}$$

Нека сега изводът да се разбие така:

$$\text{правило (1)} \quad \frac{A \to_G BC \qquad B \overset{\ell_1}{\lhd} \alpha_1 \qquad C \overset{\ell_2}{\lhd} \alpha_2}{A \overset{\ell}{\lhd} \underbrace{\alpha_1 \alpha_2}_{\alpha}} \quad (\ell = \sup\{\ell_1, \ell_2\} + 1)$$

Понеже $\ell_1 < \ell$ и $\ell_2 < \ell$, от **(И.П.)** за $R(\ell_1)$ и $R(\ell_2)$ получаваме изчислението:

$$\dfrac{\dfrac{\dfrac{A \to_G BC}{\Delta(p, \varepsilon, A) \ni (p, BC)}}{(p, \alpha_1\alpha_2, A) \vdash_P (p, \alpha_1\alpha_2, BC)} \quad \dfrac{\dfrac{B \overset{\ell_1}{\lhd} \alpha_1}{(p, \alpha_1, B) \vdash_P^\star (p, \varepsilon, \varepsilon)} \quad \dfrac{C \overset{\ell_2}{\lhd} \alpha_2}{(p, \alpha_2, C) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(p, \alpha_1\alpha_2, BC) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(p, \underbrace{\alpha_1\alpha_2}_{\alpha}, A) \vdash_P^\star (p, \varepsilon, \varepsilon)}$$

Доказателството на (б) отново ще проведем с *пълна индукция* по броя на стъпките ℓ в изчислението на стековия автомат. За целта, да разгледаме свойството

$$R(\ell) \overset{\text{деф}}{=} (\forall A \in V)(\forall \alpha \in \Sigma^\star)[\, (p, \alpha, A) \vdash^\ell (p, \varepsilon, \varepsilon) \implies A \overset{\star}{\lhd} \alpha\,].$$

Нека $(p, \alpha, A) \vdash_P^\ell (p, \varepsilon, \varepsilon)$. Имаме два варианта за първата стъпка в това изчисление.

Нека да започнем с интересния случай, който е следния:

$$\dfrac{\dfrac{\Delta(p,\varepsilon,A) \ni (p,BC)}{(p,\alpha,A) \vdash_P (p,\alpha,BC)} \quad (p,\alpha,BC) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{(p,\alpha,A) \vdash_P^{\ell} (p,\varepsilon,\varepsilon)}$$

Знаем от *Твърдение 3.7*, че можем да разбием изчислението $(p,\alpha,BC) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)$ по следния начин:

- $(p,\alpha_1,B) \vdash_P^{\ell_1} (p,\varepsilon,\varepsilon)$;
- $(p,\alpha_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon)$,

където $\alpha_1\alpha_2 = \alpha$ и $\ell_1 + \ell_2 = \ell - 1$. Понеже $\ell_1 < \ell$ и $\ell_2 < \ell$, от **(И.П.)** за $R(\ell_1)$ и $R(\ell_2)$ получаваме импликациите:

$$(p,\alpha_1,B) \vdash_P^{\ell_1} (p,\varepsilon,\varepsilon) \overset{\textbf{(И.П.)}}{\Longrightarrow} B \overset{\star}{\lhd} \alpha_1$$
$$(p,\alpha_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon) \overset{\textbf{(И.П.)}}{\Longrightarrow} C \overset{\star}{\lhd} \alpha_2.$$

Сега обединяваме всичко, което имаме и получаваме извода:

$$\dfrac{A \to_G BC \qquad \dfrac{(p,\alpha_1,B) \vdash_P^{\ell_1} (p,\varepsilon,\varepsilon)}{B \overset{\star}{\lhd} \alpha_1} \qquad \dfrac{(p,\alpha_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon)}{C \overset{\star}{\lhd} \alpha_2}}{A \overset{\star}{\lhd} \alpha}$$

Сега да разгледаме случая, когато за някое $a \in \Sigma$ имаме следното:

$$\dfrac{\dfrac{\Delta(p,\varepsilon,A) \ni (p,a)}{(p,\alpha,A) \vdash_P (p,\alpha,a)} \quad (p,\alpha,a) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)}{(p,\alpha,A) \vdash_P^{\ell} (p,\varepsilon,\varepsilon)}$$

Според конструкцията на стековия автомат трябва да имаме $\ell = 2$ и $\alpha = a$, защото това е единствения начин да имаме изчислението $(p,\alpha,a) \vdash_P^{\ell-1} (p,\varepsilon,\varepsilon)$ в нашия стеков автомат. Тук всичко е ясно, защото щом $\Delta(p,\varepsilon,A) \ni (p,a)$, то $A \to_G a$ и тогава $A \overset{\star}{\lhd} \alpha$. □

Пример 3.14. Нека е дадена безконтекстната граматика G с правила

$$S \to AS \mid BS \mid \varepsilon$$
$$A \to aA \mid a$$
$$B \to Bb \mid b.$$

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 3.7*:
  - $\Delta(q_{\texttt{start}}, \varepsilon, \sharp) = \{(q, S\sharp)\}$;
  - $\Delta(q, \varepsilon, S) = \{(q, AS), (q, BS), (q, \varepsilon)\}$;
  - $\Delta(q, \varepsilon, A) = \{(q, aA), (q, a)\}$;
  - $\Delta(q, \varepsilon, B) = \{(q, Bb), (q, b)\}$;
  - $\Delta(q, x, x) = \{(q, \varepsilon)\}$, където $x \in \{a, b\}$;
  - $\Delta(q, \varepsilon, \sharp) = \{(q_{\texttt{accept}}, \varepsilon)\}$;

Лема 3.8. 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 ако стековият автомат е в състоянието q, и стекът съдържа само променливата A, то когато стековият автомат премине в състояние p стекът ще се изпразни.

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 $(r, BC) \in \Delta(q, x, A)$, където $x \in \Sigma_\varepsilon$. Тогава за всеки две състояния q' и p добавяме правилата:

$$[q, A, p] \to_G x[r, B, q'][q', C, p].$$

- Нека имаме $(r, B) \in \Delta(q, x, A)$, където $x \in \Sigma_\varepsilon$. Тогава за всяко състояние $p \in Q$ добавяме правилата:

$$[q, A, p] \to_G x[r, B, p].$$

- Нека имаме $(p, \varepsilon) \in \Delta(q, x, A)$, където $x \in \Sigma_\varepsilon$. Тогава добавяме правилата:

$$[q, A, p] \to_G x.$$

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:

$$[q, A, p] \stackrel{\star}{\lhd} \alpha \text{ точно тогава, когато } (q, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon).$$

($\Rightarrow$) Да разгледаме свойството

Да напомним, че $\Sigma_\varepsilon \stackrel{\text{деф}}{=} \Sigma \cup \{\varepsilon\}$.

$$R(\ell) \stackrel{\text{деф}}{=} (\forall p, q \in Q)(\forall A \in V)(\forall \alpha \in \Sigma^\star)[\ (q, \alpha, A) \vdash^\ell (p, \varepsilon, \varepsilon) \implies [q, A, p] \stackrel{\star}{\lhd} \alpha\ ].$$

Ще докажем, че $(\forall \ell \in \mathbb{N})R(\ell)$. За целта, нека $(q, \alpha, A) \vdash^\ell (p, \varepsilon, \varepsilon)$. Да видим каква е първата стъпка в това изчисление.

Нека първата стъпка е получена благодарение на $(p, \varepsilon) \in \Delta(q, x, A)$. Тогава е ясно, че $\ell = 1$, $\alpha = x$ и според конструкцията на граматиката G имаме правилото $[q, A, p] \to_G x$, откъдето веднага следва, че $[q, A, p] \stackrel{1}{\lhd} x$.

Нека $\alpha = x\beta$, където $x \in \Sigma_\varepsilon$.

- Ако $\Delta(q, x, A) \ni (r, B)$, то можем да разбием изчислението по следния начин:

$$\frac{\dfrac{\Delta(q, x, A) \ni (r, B)}{(q, x\beta, A) \vdash_P (r, \beta, B)} \quad (r, \beta, B) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(q, \underbrace{x\beta}_{\alpha}, A) \vdash_P^\ell (p, \varepsilon, \varepsilon)}$$

  Сега можем да приложим индукционното предположение и да получим извода:

$$\text{(деф.)}\ \frac{\dfrac{\Delta(q, x, A) \ni (r, B)}{[q, A, p] \to_G x[r, B, p]} \quad x \in \Sigma_\varepsilon \qquad \dfrac{(r, \beta, B) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{[r, B, p] \stackrel{\star}{\lhd} \beta}\ \textbf{(и.п.)}}{[q, A, p] \stackrel{\star}{\lhd} \underbrace{x\beta}_{\alpha}}$$

- Ако $\Delta(q, a, A) \ni (r, BC)$, то можем да разбием изчислението по следния начин:

$$\frac{\dfrac{\Delta(q, a, A) \ni (r, BC)}{(q, a\beta, A) \vdash_P (r, \beta, BC)} \quad (r, \beta, BC) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(q, \underbrace{a\beta}_{\alpha}, A) \vdash_P^\ell (p, \varepsilon, \varepsilon)}$$

  Според *Твърдение 3.7*, можем да разбием β на две части като $\beta = \beta_1\beta_2$ и да получим, че:

  – $(r, \beta_1, B) \vdash_P^{\ell_1} (q', \varepsilon, \varepsilon)$;
  – $(q', \beta_2, C) \vdash_P^{\ell_2} (p, \varepsilon, \varepsilon)$,

  за някое състояние q', където $\ell_1 + \ell_2 = \ell - 1$. Понеже $\ell_1 < \ell$ и $\ell_2 < \ell$, от **(И.П.)** за $R(\ell_1)$ и $R(\ell_2)$ имаме следното:

$$(r, \beta_1, B) \vdash_P^{\ell_1} (q', \varepsilon, \varepsilon) \stackrel{\textbf{(и.п.)}}{\implies} [r, B, q'] \stackrel{\star}{\lhd} \beta_1$$
$$(q', \beta_2, C) \vdash_P^{\ell_2} (p, \varepsilon, \varepsilon) \stackrel{\textbf{(и.п.)}}{\implies} [q', C, p] \stackrel{\star}{\lhd} \beta_2.$$

Също така имаме, че $\Delta(q, x, A) \ni (r, BC)$ и според дефиницията на стековия автомат, в граматиката имаме правилото

$$[q, A, p] \to_G x[r, B, q'][q', C, p],$$

където $x \in \Sigma_\varepsilon$. Обединявайки всичко, получаваме извода в граматиката:

$$\dfrac{\dfrac{\Delta(q,x,A) \ni (r,BC)}{[q,A,p] \to_G x[r,B,q'][q',C,p]} \quad x \in \Sigma_\varepsilon \quad \dfrac{(r,\beta_1,B) \vdash_P^{\ell_1} (q',\varepsilon,\varepsilon)}{[r,B,q'] \overset{\star}{\lhd} \beta_1} \quad \dfrac{(q',\beta_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon)}{[q',C,p] \overset{\star}{\lhd} \beta_2}}{[q,A,p] \overset{\star}{\lhd} \underbrace{x\beta_1\beta_2}_{\alpha}}$$

($\Leftarrow$) За тази посока, да разгледаме свойството

$$R(\ell) \overset{\text{деф}}{=} (\forall q,p \in Q)(\forall A \in V)(\forall \alpha \in \Sigma^\star)[\, [q,A,p] \overset{\ell}{\lhd} \alpha \implies (q,\alpha,A) \vdash^\star (p,\varepsilon,\varepsilon)\,].$$

Ще докажем с пълна индукция, че $(\forall \ell \in \mathbb{N})R(\ell)$. Нека $[q, A, p] \overset{\ell}{\lhd} \alpha$. Да видим как е получен този извод.

- Първият случай е следния:

$$\dfrac{[q,A,p] \to_G x}{[q,A,p] \overset{1}{\lhd} \underbrace{x}_{\alpha}}$$

  Според дефиницията на граматиката, правилото $[q, A, p] \to_G x$ е добавено към граматиката, защото в стековият автомат имаме $\Delta(q, x, A) \ni (p, \varepsilon)$. Тогава е ясно, че $(q, x, A) \vdash_P (p, \varepsilon, \varepsilon)$.

Тук отново е възможно $x = \varepsilon$. Това не е проблем, защото правим индукция по дължината на извода, а не по дължината на думата α.

- Второ, да приемем, че имаме следния извод:

$$\dfrac{[q,A,p] \to_G x[r,B,p] \quad x \in \Sigma_\varepsilon \quad [r,B,p] \overset{\ell-1}{\lhd} \beta}{[q,A,p] \overset{\ell}{\lhd} \underbrace{x\beta}_{\alpha}}$$

  Понеже $\ell - 1 < \ell$, от **(И.П.)** за $R(\ell - 1)$ получаваме импликацията:

$$[r,B,p] \overset{\ell-1}{\lhd} \beta \overset{\textbf{(И.П.)}}{\implies} (r,\beta,B) \vdash^\star (p,\varepsilon,\varepsilon).$$

  Сега можем да приложим индукционното предположение и да сглобим изчислението:

$$\dfrac{\dfrac{\dfrac{[q,A,p] \to_G x[r,B,p]}{\Delta(q,x,A) \ni (r,B)}\ (\text{деф. на } G)}{(q,x\beta,A) \vdash_P (r,\beta,B)} \quad \dfrac{[r,B,p] \overset{\ell-1}{\lhd} \beta}{(r,\beta,B) \vdash_P^\star (p,\varepsilon,\varepsilon)}\ \textbf{(И.П.)}}{(q,\underbrace{x\beta}_{\alpha},A) \vdash_P^\star (p,\varepsilon,\varepsilon)}$$

- Трето, да разгледаме случая:

$$\frac{[q,A,p] \to_G x[r,B,q'][q',C,p] \quad x \in \Sigma_\varepsilon \quad [r,B,q'] \overset{\ell_1}{\lhd} \beta_2 \quad [q',C,p] \overset{\ell_2}{\lhd} \beta_2}{[q,A,p] \overset{\ell}{\lhd} \underbrace{x\beta_1\beta_2}_{\alpha}}$$

Понеже $\ell = 1 + \sup\{\ell_1, \ell_2\}$, то $\ell_1 < \ell$ и $\ell_2 < \ell$, от **(И.П.)** за $R(\ell_1)$ и $R(\ell_2)$ получаваме импликациите:

$$[r,B,q'] \overset{\ell_1}{\lhd} \beta_1 \overset{\textbf{(И.П.)}}{\Longrightarrow} (r,\beta_1,B) \vdash_P^\star (q',\varepsilon,\varepsilon)$$
$$[q',C,p] \overset{\ell_2}{\lhd} \beta_2 \overset{\textbf{(И.П.)}}{\Longrightarrow} (q',\beta_2,C) \vdash_P^\star (p,\varepsilon,\varepsilon).$$

Правилото $[q,A,p] \to_G x[r,B,q'][q',C,p]$ е добавено в граматиката, защото $\Delta(q,x,A) \ni (r,BC)$. Обединявайки всичко, което знаем, получаваме изчислението:

$$\frac{\dfrac{\dfrac{[q,A,p] \to_G x[r,B,q'][q',C,p]}{\Delta(q,x,A) \ni (r,BC)}}{(q,x\beta_1\beta_2,A) \vdash_P (r,\beta_1\beta_2,BC)} \quad \dfrac{\dfrac{[r,B,q'] \overset{\ell_1}{\lhd} \beta_1}{(r,\beta_1,B) \vdash_P^\star (q',\varepsilon,\varepsilon)} \quad \dfrac{[q',C,p] \overset{\ell_2}{\lhd} \beta_2}{(q',\beta_2,C) \vdash_P^\star (p,\varepsilon,\varepsilon)}}{(r,\beta_1\beta_2,BC) \vdash_P^\star (p,\varepsilon,\varepsilon)}}{(q,\underbrace{x\beta_1\beta_2}_{\alpha},A) \vdash_P^\star (p,\varepsilon,\varepsilon)}$$

□

Пример 3.15. Да разгледаме стековия автомат от Пример 3.13.

- 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 3.7. Класът на езиците, които се разпознават от недетерминирани стекови автомати съвпада с класа на безконтекстните езици.

Сечение с регулярен език

От *Твърдение 3.6* 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 3.8. Нека L е безконтекстен език и R е регулярен език. Тогава тяхното сечение $L \cap R$ е безконтекстен език.

Тук адаптираме доказателството от [18, стр. 144].

Упътване. Нека имаме стеков автомат

$$P = \langle Q', \Sigma, \Gamma, \natural, \Delta', q'_{\texttt{start}}, q'_{\texttt{accept}} \rangle, \text{ където } \mathcal{L}(P) = L,$$

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 2.3*.

Ще определим нов стеков автомат $\mathcal{M} = \langle Q, \Sigma, \Gamma, \natural, \Delta, q_{\texttt{start}}, q_{\texttt{accept}} \rangle$, където:

- $Q \stackrel{\text{деф}}{=} Q' \times Q''$;
- $q_{\texttt{start}} \stackrel{\text{деф}}{=} \langle q'_{\texttt{start}}, q''_{\texttt{start}} \rangle$;
- $F \stackrel{\text{деф}}{=} \{q'_{\texttt{accept}}\} \times F''$;
- 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 $\mathcal{L}(\mathcal{M}) = \mathcal{L}(P) \cap \mathcal{L}(\mathcal{A})$!
- 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

  $$(\langle q_1, q_2 \rangle, \alpha, \gamma) \vdash^\star_{\mathcal{M}} (\langle p_1, p_2 \rangle, \varepsilon, \varepsilon).$$

□

Упътване. Можем да построим безконтекстна граматика G' директно по безконтекстна граматика G в нормална форма на Чомски и ДКА $\mathcal{A}$. Правилата на G' са следните: [4, стр. 32]

- $[q, A, p] \to_{G'} [q, C, r][r, B, p]$, ако $A \to_G BC$ и $q, r, p \in Q_{\mathcal{A}}$;
- $[q, A, p] \to_{G'} a$, ако $A \to_G a$ и $\delta_{\mathcal{A}}(q, a) = p$;
- $S' \to_{G'} [q_{\texttt{start}}, S, f]$, за всяко $f \in F_{\mathcal{A}}$.

Докажете, че $[q, A, p] \overset{\star}{\lhd}_{G'} \alpha$ точно тогава, когато $A \overset{\star}{\lhd}_G \alpha$ и $\delta^{\star}_{\mathcal{A}}(q, \alpha) = p$.

□

Теорема 3.8 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 3.16. 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 3.8*, $L' = L \cap \mathcal{L}[\![\boldsymbol{a^{\star}b^{\star}c^{\star}}]\!]$ също е безконтекстен език. Но $L' = \{a^n b^n c^n \mid n \in \mathbb{N}\}$, за който знаем от *Пример 3.5*, че *не* е безконтекстен. Достигнахме до противоречие. Следователно, L не е безконтекстен език.

3.7 Допълнителни задачи

3.7.1 Равен брой леви и десни скоби

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 }.

Нека за по-голяма яснота да положим

$$\texttt{left}(\alpha) \stackrel{\text{деф}}{=} |\alpha|_a \qquad \text{// брой срещания на } a \text{ в } \alpha$$

$$\texttt{right}(\alpha) \stackrel{\text{деф}}{=} |\alpha|_b \qquad \text{// брой срещания на } b \text{ в } \alpha$$

Задача 3.22. 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:

- $\omega = \omega_1 b \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

Другият случай е аналогичен

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.

- $|\omega| = 1$. 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 $\omega = b^i ba\omega'$, 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 3.23. 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 3.24. Да се докаже, че езикът $L = \{\ \alpha \in \{a,b\}^\star \mid \texttt{left}(\alpha) = \texttt{right}(\alpha)\ \}$ е безконтекстен.

Доказателство. Една възможна граматика G е следната:

$$S \to aSbS \mid bSaS \mid \varepsilon.$$

Алтернативна граматика за езика L е

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

Като следствие от *Задача 3.22* 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 $\mathcal{L}_G^\ell(S) = \{\omega \in \Sigma^\star \mid S \overset{\leq \ell}{\lhd} \omega\}$ и според *Твърдение 3.2* имаме следната рекурсивна връзка:

$$\begin{aligned}
\mathcal{L}_G^0(S) =& \emptyset \\
\mathcal{L}_G^{\ell+1}(S) =& \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{b\} \cdot \mathcal{L}_G^\ell(S) \cup \\
& \{b\} \cdot \mathcal{L}_G^\ell(S) \cdot \{a\} \cdot \mathcal{L}_G^\ell(S) \cup \\
& \{\varepsilon\}.
\end{aligned}$$

За коректност на граматиката трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}_G^\ell(S) \subseteq \{\omega \in \{a,b\}^\star \mid \texttt{left}(\omega) = \texttt{right}(\omega)\}. \tag{3.21}$$

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 $\omega \in \mathcal{L}_G^{\ell+1}(S)$. 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 3.22* 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:

$$\{\omega \in \{a,b\}^\star \mid \texttt{left}(\omega) = \texttt{right}(\omega)\} \subseteq \mathcal{L}_G(S).$$

□

3.7.2 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 3.25. Докажете, че $L = \{\ \alpha \in \{a,b\}^\star \mid \alpha$ е балансирана дума $\}$ е безконтекстен език.

Доказателство. Да разгледаме граматиката G с правила

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

[13, стр. 135]
✍ Докажете, че езикът L не е регулярен!
Алтернативна граматика е

$$S \to aSbS \mid \varepsilon.$$

Ще докажем, че $L = \mathcal{L}(G)$. Имаме, че

$$\begin{aligned}
\mathcal{L}^0_G(S) =& \emptyset \\
\mathcal{L}^{\ell+1}_G(S) =& \{a\} \cdot \mathcal{L}^\ell_G(S) \cdot \{b\} \cup \\
& \mathcal{L}^\ell_G(S) \cdot \mathcal{L}^\ell_G(S) \cup \\
& \{\varepsilon\}.
\end{aligned}$$

Първо да разгледаме коректност на граматиката, т.е. трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}^\ell_G(S) \subseteq \{\ \alpha \in \{a,b\}^\star \mid \alpha \text{ е балансирана дума } \}. \tag{3.23}$$

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 $\omega \in \mathcal{L}^{\ell+1}_G(S)$. Имаме три случая.

- Нека $\omega = \varepsilon$. Тогава е ясно, че ω е балансирана дума.
- Нека $\omega \in \{a\} \cdot \mathcal{L}^\ell_G(S) \cdot \{b\}$. Тогава $\omega = a\omega_1 b$ и $\omega_1 \in \mathcal{L}^\ell_G(S)$. От **(И.П.)** имаме, че ω_1 е балансирана. Лесно се съобразява, че ω също е балансирана.
- Нека $\omega \in \mathcal{L}^\ell_G(S) \cdot \mathcal{L}^\ell_G(S)$. Тогава $\omega = \omega_1\omega_2$, такива че $\omega_1, \omega_2 \in \mathcal{L}^\ell_G(S)$. От **(И.П.)** имаме, че ω_1 и ω_2 са балансирани. Лесно се съобразява, че ω също е балансирана.

Така доказахме, че

$$\mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}^\ell_G(S) \subseteq \{\ \alpha \in \{a,b\}^\star \mid \alpha \text{ е балансирана дума } \}.$$

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.

  Тук $\omega_1 \neq \varepsilon$ и $\omega_1 \neq \alpha$. Например, ab е същински префикс на $abab$, който е балансирана дума.

  – 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 $\omega \in \mathcal{L}_G(S)$.

  Например, $aabb$ няма същински префикс, който да 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.

    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. □

3.7.3 Лесни задачи

Задача 3.26. 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 3.27. Докажете, че следните езици са безконтекстни.

а) $L = \{\omega \in \{a,b\}^\star \mid \omega = \omega^{\mathtt{rev}}\}$;

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) $L = \{a^n b^k c^m \mid n + k + 1 \geq m\}$;

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 3.28. 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) $\{a^n b^n c^m \mid m \leq n\}$;

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) $S \to EaE$,
$E \to aEbE \mid bEaE \mid \varepsilon$;

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) $\{\omega \in \{a,b,c\}^\star \mid |\omega|_a \neq |\omega|_b \vee |\omega|_a \neq |\omega|_c \vee |\omega|_b \neq |\omega|_c\}$.

3.7.4 Не толкова лесни задачи

Задача 3.29. 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:

$$\texttt{diff}(\alpha, \varepsilon) = 0$$
$$\texttt{diff}(\varepsilon, \beta) = 0$$
$$\texttt{diff}(a \cdot \alpha, b \cdot \beta) = \begin{cases} \texttt{diff}(\alpha, \beta), & \text{ако } a = b \\ 1 + \texttt{diff}(\alpha, \beta), & \text{ако } a \neq b \end{cases}$$

Задача 3.30. 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 3.31. 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

$$D_1 = L_1 \cdot (\sharp L_1)^\star \cdot (\{\varepsilon\} \cup \sharp\{a,b\}^\star) \cap$$
$$\{a,b\}^\star \cdot (\sharp L_1)^\star \cdot (\{\varepsilon\} \cup \sharp\{a,b\}^\star).$$

□

Задача 3.32. 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 3.33. Нека L_1 и L_2 са езици. Дефинираме [25, 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 3.34. Докажете, че ако L е безконтекстен език, то

$$L^{\texttt{rev}} = \{\omega^{\texttt{rev}} \mid \omega \in L\}$$

също е безконтекстен.

Задача 3.35. 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 3.36. 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 3.37. 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 3.38. 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 3.39. 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 3.40. Докажете, че всеки безконтекстен език над азбуката $\Sigma = \{a\}$ е регулярен.

Задача 3.41. 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 3.42. 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 3.43. 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 3.44. 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. □

Триене на третина от думите на език

Да напомним следните операции върху езици:

$$\texttt{Triples}_2(L) \stackrel{\text{деф}}{=} \{\omega_2 \mid (\exists\omega_1)(\exists\omega_3)[\,\omega_1\omega_2\omega_3 \in L \;\&\; |\omega_1| = |\omega_2| = |\omega_3|\,]\,\};$$

$$\texttt{Triples}_{1,3}(L) \stackrel{\text{деф}}{=} \{\omega_1\omega_3 \mid (\exists\omega_2)[\omega_1\omega_2\omega_3 \in L \;\&\; |\omega_1| = |\omega_2| = |\omega_3|]\}.$$

Вече знаем, че $\texttt{Triples}_{1,3}(L)$ не винаги е регулярен при регулярен L. Възможно за регулярен L да се конструира и стеков автомат [7, стр. 140].

Задача 3.45. Докажете, че ако регулярен език L, то $\texttt{Triples}_{1,3}(L)$ е безконтекстен.

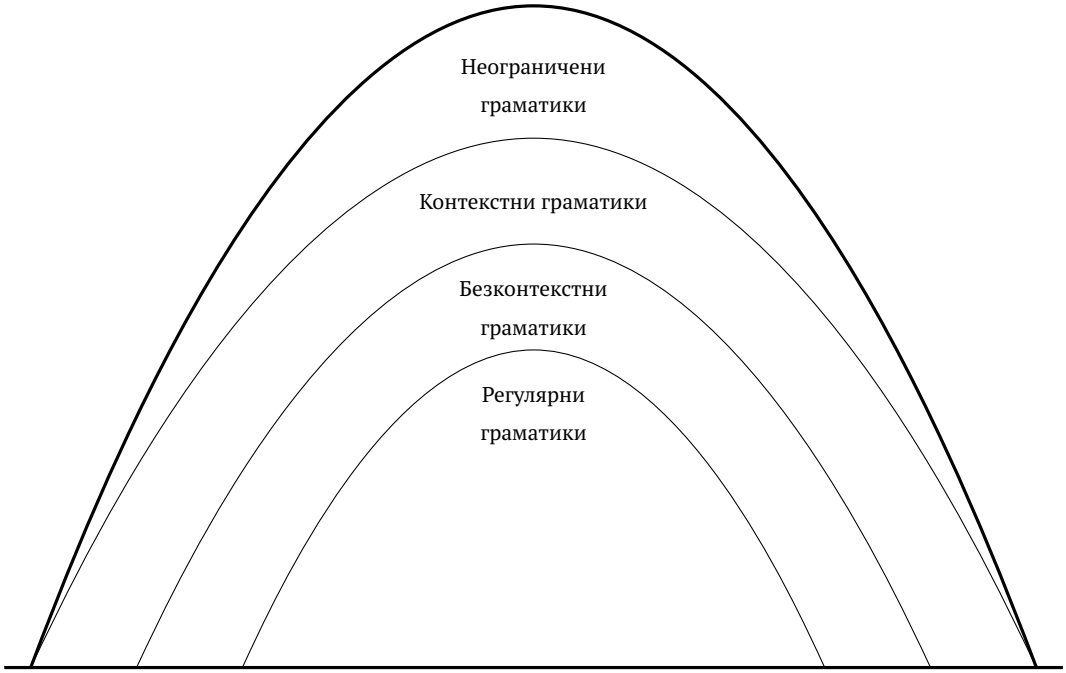
Упътване. Нека $L = \mathcal{L}(\mathcal{A})$, където $\mathcal{A}$ е ДКА. Ще построим безконтекстна граматика G за $\texttt{Triples}_{1,3}(L)$. Правилата на граматиката G са следните:

- $S \to [(q_{\texttt{start}}, q), (q, r), f]$ за произволно $f \in F$.
- $[(p, q), (r, s), t] \to a[(p', q), (r', s), t']b$, където са изпълени условията:
  - $\delta_{\mathcal{A}}(p, a) = p'$;
  - $\delta_{\mathcal{A}}(t', b) = t$, за някое t';
  - $\delta_{\mathcal{A}}(r, x) = r'$ за някое $x \in \Sigma$.
- $[(p, p), (q, q), q] \to \varepsilon$ за произволни $p, q \in Q$.

□

Глава 4

Йерархия от езици



4.1 Неограничени граматики

На англ. *unrestricted grammar*. Това са тип 0 граматиките в йерархията на Чомски [10, 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 [10] правилата се наричат *productions* или *production rules*.

- $S \in V$ е началната променлива (нетерминал).

Ще дефинираме бинарната релация $\Rightarrow_G$ над $(V \cup \Sigma)^\star$, така че $\gamma \Rightarrow_G \gamma'$ казва, че от думата γ се получава думата γ' чрез прилагане на някое правило $\alpha \to_G \beta$ в граматиката G.

Тук отново, когато е ясно за коя граматика говорим, ще пишем просто $\Rightarrow$ вместо $\Rightarrow_G$.

$$\frac{(\alpha, \beta) \in R \qquad \lambda, \rho \in (V \cup \Sigma)^\star}{\lambda\alpha\rho \Rightarrow_G \lambda\beta\rho}$$

Фигура 4.1: Дефиниция на релацията $\Rightarrow_G$, която казва дали от една дума може да се получи друга дума с прилагане на едно правило от граматиката.

Задача 4.1. 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 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 4.2: Правила за извод в неограничена граматика

Пример 4.1. Да разгледаме граматиката G зададена с правилата

$$S \to_G SA \mid aSA \mid ASb \mid \varepsilon.$$

Тогава $aSbSb \Rightarrow_G aaSbbSb$, защото имаме правилото $S \to_G aSb$. Също така, $aSSb \not\Rightarrow_G ab$, но $aSSb \overset{2}{\Rightarrow}_G ab$ като приложим два пъти правилото $S \to_G \varepsilon$.

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 $\Rightarrow_G$.

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 4.1. 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:

$$\text{(}\textit{Задача 4.1}\text{)}\ \frac{\dfrac{\alpha \Rightarrow \gamma}{\lambda\alpha\rho \Rightarrow \lambda\gamma\rho} \qquad \dfrac{\gamma \overset{\ell-1}{\Rightarrow} \beta}{\lambda\gamma\rho \overset{\ell-1}{\Rightarrow} \lambda\beta\rho}\ \textbf{(И.П.)}}{\lambda\alpha\rho \overset{\ell}{\Rightarrow} \lambda\beta\rho}\ \text{(транзитивност)}$$

□

Твърдение 4.2. 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:

$$\text{(\textit{Твърдение 4.1})}\ \frac{\alpha_1 = \beta_1 \qquad \alpha_2 \overset{\ell_2}{\Rightarrow}_G \beta_2}{\alpha_1\alpha_2 \overset{\ell_2}{\Rightarrow}_G \beta_1\beta_2}$$

- 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.)** за $\ell_1 - 1 < \ell_1$ и получаваме следния извод:

$$\text{(\textit{Задача 4.1})}\ \frac{\dfrac{\alpha_1 \Rightarrow \gamma_1}{\alpha_1\alpha_2 \Rightarrow \gamma_1\alpha_2} \qquad \dfrac{\gamma_1 \overset{\ell_1-1}{\Rightarrow} \beta_1 \qquad \alpha_2 \overset{\ell_2}{\Rightarrow} \beta_2}{\gamma_1\alpha_2 \overset{\ell_1-1+\ell_2}{\Rightarrow} \beta_1\beta_2}\ \textbf{(И.П.)}}{\alpha_1\alpha_2 \overset{\ell_1+\ell_2}{\Rightarrow} \beta_1\beta_2}\ \text{(транзитивност)}$$

□

Твърдение 4.3. 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 4.4. 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.

- Нека $\ell_1 = 0$. Този случай е ясен, защото тогава $\alpha = \beta$ и имаме извода:

$$\dfrac{\dfrac{\alpha \overset{0}{\Rightarrow} \beta}{\alpha = \beta} \qquad \beta \overset{\ell_2}{\Rightarrow} \gamma}{\alpha \overset{0+\ell_2}{\Rightarrow} \gamma} \text{ (рефлексивност)}$$

- Нека $\ell_1 > 0$. 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 4.1. За произволни естествени числа ℓ_1 и ℓ_2 е изпълнено, че:

$$\frac{\alpha \overset{\ell_1}{\Rightarrow} \lambda\beta\rho \qquad \beta \overset{\ell_2}{\Rightarrow} \gamma}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow} \lambda\gamma\rho}$$

4.2 Контекстни граматики

На англ. context-sensitive [10, 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 [24, 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 4.2. 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 изпълнено 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 имаме включването $L \subseteq \mathcal{L}(G)$. □

4.3 Безконтекстни граматики

В Раздел 4.1 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 Фигура 4.1 в частния случай, когато граматиката е безконтекстна.

В [18] 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:

$$\frac{(A, \alpha) \in R}{A \Rightarrow_G \alpha}$$

$$\frac{(A, \alpha) \in R \qquad \lambda, \rho \in (V \cup \Sigma)^\star}{\lambda A \rho \Rightarrow_G \lambda \alpha \rho}$$

Фигура 4.3: Дефиниция на релацията $\Rightarrow_G$, която казва дали от една дума може да се получи друга дума с прилагане на едно правило от граматиката.

Нека официално да обявим, че един език L се нарича **безконтекстен**, ако съществува безконтекстна граматика G, за която

$$L = \mathcal{L}(G) = \{\omega \in \Sigma^\star \mid S \overset{\star}{\Rightarrow} \omega\}.$$

Да започнем с няколко прости примера за езици, които са безконтекстни, но за които знаем, че не са регулярни.

От тези прости примери можем да заключим, че безконтекстните граматики имат странни навици в сравнение с автоматите. Докато автоматите ядат първо супата, после основното и накрая десерта, то безконтекстните граматики ядат супата заедно с десерта.

Пример 4.3. Езикът $\{a^n b^n \mid n \in \mathbb{N}\}$ е безконтекстен, защото може да се опише с граматика с правила $S \to aSb \mid \varepsilon$.

Пример 4.4. Езикът $\{\omega\omega^{\mathtt{rev}} \mid \omega \in \{a, b\}^\star\}$ е безконтекстен, защото може да се опише с граматика с правила $S \to aSa \mid bSb \mid \varepsilon$.

По-късно ще видим как можем внимателно да докажем че езиците от примерите наистина се пораждат (генерират) от дадените граматики.

Нека сега да обърнем внимание на някои общи свойства на безконтекстните граматики. Като частен случай на *Следствие 4.1* получаваме следното свойство.

Твърдение 4.5. За произволни естествени числа ℓ_1 и ℓ_2 е изпълнено, че:

$$\frac{\alpha \overset{\ell_1}{\Rightarrow} \lambda B \rho \qquad B \overset{\ell_2}{\Rightarrow} \beta}{\alpha \overset{\ell_1+\ell_2}{\Rightarrow} \lambda \beta \rho,}$$

Пример 4.5. Да разгледаме безконтекстната граматика G, която има следните правила:

$$S \to_G AS \mid \varepsilon$$
$$A \to_G aAb \mid ab.$$

За момента не е ясно как можем да проверим, че примерно думата $abbba \notin \mathcal{L}(G)$. Този въпрос ще разгледаме по-нататък.

Да видим защо думата $aabbab \in \mathcal{L}(G)$. Ако следваме формално правилата за извод, получаваме следното:

$$\dfrac{\dfrac{\dfrac{(S,AS) \in R}{S \Rightarrow_G AS} \quad \dfrac{\dfrac{(S,\varepsilon) \in R}{S \Rightarrow_G \varepsilon} \quad \dfrac{}{\varepsilon \overset{0}{\Rightarrow}_G \varepsilon}}{S \overset{1}{\Rightarrow}_G \varepsilon}\ (\textit{Тв. 4.5}) \quad \dfrac{\dfrac{(S,AS) \in R}{S \Rightarrow_G AS} \quad \dfrac{}{AS \overset{0}{\Rightarrow}_G AS}}{S \overset{1}{\Rightarrow}_G AS}\ (\textit{Тв. 4.5})}{S \overset{2}{\Rightarrow}_G A}\ (\textit{Тв. 4.2}) \quad \dfrac{}{A \overset{0}{\Rightarrow}_G A}}{AS \overset{2}{\Rightarrow}_G AA}\ (\text{транзитивност})}{S \overset{3}{\Rightarrow}_G AA}$$

Освен това имаме и следния формален извод:

$$\dfrac{\dfrac{(A,aAb) \in R}{A \Rightarrow_G aAb} \quad \dfrac{(A,ab) \in R}{aAb \overset{1}{\Rightarrow}_G aabb}}{A \overset{2}{\Rightarrow}_G aabb}\ (\text{транзитивност})$$

Аналогично,

$$\dfrac{\dfrac{(A,ab) \in R}{A \Rightarrow_G ab} \quad \dfrac{}{ab \overset{0}{\Rightarrow}_G ab}}{A \overset{1}{\Rightarrow}_G ab}$$

Обединявайки всичко, получаваме:

$$\dfrac{\dfrac{S \overset{3}{\Rightarrow}_G AA \quad A \overset{2}{\Rightarrow}_G aabb}{S \overset{5}{\Rightarrow}_G aabbA}\ (\textit{Тв. 4.5}) \quad A \overset{1}{\Rightarrow}_G ab}{S \overset{6}{\Rightarrow}_G aabbab}\ (\textit{Тв. 4.5})$$

Можем да приложим правилата за извод в различен ред и пак да получим същия краен резултат. Например:

$$\dfrac{\dfrac{\dfrac{\dfrac{(S,AS) \in R}{S \Rightarrow_G AS} \quad \dfrac{A \overset{2}{\Rightarrow} aabb}{AS \overset{2}{\Rightarrow} aabbS}\ (\textit{Тв. 4.1})}{S \overset{3}{\Rightarrow}_G aabbS} \quad S \overset{2}{\Rightarrow}_G A}{S \overset{5}{\Rightarrow}_G aabbA}\ (\textit{Тв. 4.5}) \quad A \overset{1}{\Rightarrow}_G ab}{S \overset{6}{\Rightarrow}_G aabbab}\ (\textit{Тв. 4.5})$$

Естествено, ние няма да правим на ръка такива формални изводи в граматика, но е важно да придобием опит с механиката, чрез която се извършва един извод. Следващото твърдение ни дава едно важно свойство, което е вярно за безконтекстни граматики, но не и за неограничени граматики.

Лема 4.1 (Лема за разбиване на извода)**.** 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.

Тук $\gamma_1, \gamma_2, \beta \in (V \cup \Sigma)^\star$.

- Нека $\ell = 0$. Тогава $\beta_1 = \gamma_1$ и $\beta_2 = \gamma_2$ и очевидно $\ell_1 = \ell_2 = 0$.
- Нека $\ell > 0$. Тогава да разгледаме следната ситуация:

$$\dfrac{\dfrac{(A,\alpha) \in R}{\lambda A\rho \Rightarrow_G \lambda\alpha\rho} \quad \lambda\alpha\rho \overset{\ell-1}{\Rightarrow} \beta}{\underbrace{\lambda A\rho}_{\gamma_1\gamma_2} \overset{\ell}{\Rightarrow} \beta}$$

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 можем да запишем думите γ_1 и γ_2 като $\gamma_1 = \lambda A\rho_1$ 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 4.6. 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 *Лема 4.1*. □

Тук е възможно $X_i = a \in \Sigma$. Тогава $a \overset{0}{\Rightarrow} a$ и $\beta_i = a$.

Лема 4.2. 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

$$\frac{X \to_G X_1 \cdots X_k \qquad X_1 \cdots X_k \overset{\ell-1}{\Rightarrow} \beta}{X \overset{\ell}{\Rightarrow} \beta}\ \text{(транз.)}$$

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 4.6* 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:

$$\frac{X \to_G X_1 \cdots X_k \qquad \dfrac{X_1 \overset{\ell_1}{\Rightarrow} \beta_1}{X_1 \overset{\star}{\lhd} \beta_1}\ \textbf{(и.п.)} \qquad \cdots \qquad \dfrac{X_k \overset{\ell_k}{\Rightarrow} \beta_k}{X_k \overset{\star}{\lhd} \beta_k}\ \textbf{(и.п.)}}{X \overset{\star}{\lhd} \underbrace{\beta_1 \cdots \beta_k}_{\beta}}\ \text{правило (1)}$$

□

Лема 4.3. 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:

$$\frac{X \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \gamma_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \gamma_n}{X \overset{\ell}{\lhd} \underbrace{\gamma_1 \cdots \gamma_n}_{\gamma}}\ (\ell = 1 + \sup\{\ell_1, \ldots, \ell_n\})$$

Понеже $\ell_i < \ell$ за всяко $i = 1, \ldots, n$, получаваме следното:

$$\dfrac{X \to_G X_1 \cdots X_n \qquad \dfrac{\dfrac{X_1 \overset{\ell_1}{\lhd} \gamma_1}{X_1 \overset{\star}{\Rightarrow} \gamma_1} \text{(И.П.)} \quad \cdots \quad \dfrac{X_n \overset{\ell_n}{\lhd} \gamma_n}{X_n \overset{\star}{\Rightarrow} \gamma_n} \text{(И.П.)}}{X_1 \cdots X_n \overset{\star}{\Rightarrow} \gamma_1 \cdots \gamma_n} \text{(Тв. 4.3)}}{X \overset{\star}{\Rightarrow} \underbrace{\gamma_1 \cdots \gamma_n}_{\gamma}} \text{(транз.)}$$

□

Комбинирайки предишните две леми получаваме следната теорема.

Теорема 4.1. 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 $X \overset{\star}{\Rightarrow} \gamma$.

Следващото твърдение е важно, но е трудно да не пропуснем доказателството му.

Твърдение 4.7. 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 4.8. 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.

□

4.4 Регулярни граматики

Също така се наричат граматики от тип 3 в йерархията на Чомски [10, стр. 217]. Този вид граматики понякога се нарича и дясно-регулярна граматика.

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 $a \in \Sigma$.

Лема 4.4. 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:

- $Q \stackrel{\text{деф}}{=} \{q_0, \ldots, q_k\}$;
- $Q_{\texttt{start}} \stackrel{\text{деф}}{=} \{q_0\}$;
- $F \stackrel{\text{деф}}{=} \{q_i \mid A_i \to \varepsilon\}$;
- $\Delta(q_i, a) \stackrel{\text{деф}}{=} \{q_j \mid A_i \to aA_j$ е правило в граматиката$\}$.

Докажете, че $\mathcal{L}(\mathcal{N}) = \mathcal{L}(G)$. □

Лема 4.5. За всеки детерминиран краен автомат $\mathcal{A}$ съществува регулярна граматика G, такава че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(G)$.

Тази конструкция може да се приложи и за недетерминиран автомат. Единствено трябва да се внимава, ако автоматът има много начални състояния.

Упътване.

Използваме *Твърдение 2.2*.

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:

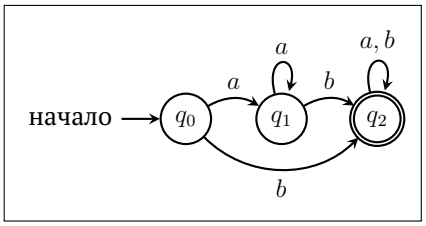
- $V \stackrel{\text{деф}}{=} \{A_0, \ldots, A_k\}$;
- $S \stackrel{\text{деф}}{=} A_0$;
- $A_i \to aA_j \stackrel{\text{деф}}{\Leftrightarrow} \delta(q_i, a) = q_j$;

- $A_i \to \varepsilon \overset{\text{деф}}{\Leftrightarrow} q_i \in F$.

Докажете, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(G)$. □

Теорема 4.2. Един език е регулярен точно тогава, когато се поражда от регулярна граматика.

Пример 4.6. Да разгледаме отново автомата от *Фигура 2.21*.



Фигура 4.4: $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbf{a^*b(a+b)^*})$.

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 4.2. 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 4.3. 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.

Глава 5

Машини на Тюринг

> Turing's 'Machines'. These machines are humans who calculate. [26, § 1096].

Детерминирана 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 [25, 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 [25, стр. 169] и [9, 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.

Причината за да искаме да имаме $\sqcup$ след думата α за да може лесно да работим със случая, когато $\alpha = \varepsilon$.

- **Началната конфигурация** за входната дума $\alpha \in \Sigma^\star$ представлява тройката

$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup).$$

- **Приемаща конфигурация** представлява тройка от вида

$$(\lambda, q_{\texttt{accept}}, \rho).$$

- **Отхвърляща конфигурация** представлява тройка от вида

$$(\lambda, q_{\texttt{reject}}, \rho).$$

- Една конфигурация ще наричаме **заключителна**, ако тя е или приемаща или отхвърляща.

Както за автомати, удобно е да дефинираме бинарна релация $\vdash_{\mathcal{M}}$ над множеството $\Gamma^\star \times Q \times \Gamma^+$, която ще казва как моментната конфигурация на машината $\mathcal{M}$ се променя при изпълнение на една стъпка.

За да направим това, удобно е първо да дефинираме бинарната релация $\vdash_{y,d}$ над множеството $\Gamma^\star \times \Gamma^+$, която показва как една моментна конфигурация се променя, когато заменим символа на главата с y и се придвижим на посока $d \in \{\lhd, \rhd, \square\}$.

Дефиницията на релацията $\vdash_{y,d}$ не зависи от конкретна машина на Тюринг!

$$\frac{x,y,z \in \Gamma \qquad \lambda,\rho \in \Gamma^\star}{(\lambda, xz\rho) \vdash_{y,\triangleright} (\lambda y, z\rho)} \qquad \frac{x,y \in \Gamma \qquad \lambda \in \Gamma^\star}{(\lambda, x) \vdash_{y,\triangleright} (\lambda y, \sqcup)}$$

$$\frac{x,y,z \in \Gamma \qquad \lambda,\rho \in \Gamma^\star}{(\lambda z, x\rho) \vdash_{y,\triangleleft} (\lambda, zy\rho)} \qquad \frac{x,y \in \Gamma \qquad \rho \in \Gamma^\star}{(\varepsilon, x\rho) \vdash_{y,\triangleleft} (\varepsilon, \sqcup y\rho)}$$

$$\frac{x,y \in \Gamma \qquad \lambda,\rho \in \Gamma^\star}{(\lambda, x\rho) \vdash_{y,\square} (\lambda, y\rho)}$$

Фигура 5.1: Дефиниция на релацията $\vdash_{y,d}$.

Сега вече сме готови да дефинираме релацията $\vdash_{\mathcal{M}}$.

Ако няма опасност да се заблудим за коя точно машина на Тюринг $\mathcal{M}$ говорим, то е възможно да пишем просто $\vdash$ вместо $\vdash_{\mathcal{M}}$.

$$\frac{\delta(q,x) = (q', y, d) \qquad (\lambda, x\rho) \vdash_{y,d} (\lambda', \rho')}{(\lambda, q, x\rho) \vdash_{\mathcal{M}} (\lambda', q', \rho')}$$

Фигура 5.2: Преход в еднолентова детерминистична машина на Тюринг $\mathcal{M}$

Сега за всяко естествено число ℓ, ще дефинираме релацията $\vdash^{\ell}_{\mathcal{M}}$, която ще казва, че от конфигурацията κ можем да достигнем до конфигурацията κ' за ℓ на брой стъпки.

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

- С $\vdash^{\star}_{\mathcal{M}}$ ще означаваме рефлексивното и транзитивно затваряне на релацията $\vdash$ или с други думи,

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

- Макар и една конфигурация κ да преставлява тройка, то често ще бъде удобно да гледаме на κ като на дума от езика $\Gamma^\star Q \Gamma^+$.
- Важно свойство е, че ако $\kappa \vdash^{\star}_{\mathcal{M}} \kappa'$, 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 $\lambda, \rho \in \Gamma^\star$,

$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash_{\mathcal{M}}^\star (\lambda, q_{\texttt{reject}}, \rho).$$

- Машината на Тюринг $\mathcal{M}$ **не приема** думата α, ако $\mathcal{M}$ отхвърля α или $\mathcal{M}$ никога не завършва при начална конфигурация $(\varepsilon, q_{\texttt{start}}, \alpha)$.

На англ. такава машина на Тюринг се нарича **decider** [25, стр. 170]. Може такива машини на Тюринг да се наричат и тотални [13, стр. 213]. Да се внимава, че в Манев понятията са различни.

- Една машина на Тюринг се нарича **разрешител**, ако при всеки вход достига до заключително състояние, т.е. достига до $q_{\texttt{accept}}$ или $q_{\texttt{reject}}$.

- Езикът, който се **разпознава** от машината $\mathcal{M}$ е:

$$\mathcal{L}(\mathcal{M}) \stackrel{\text{деф}}{=} \{\alpha \in \Sigma^\star \mid (\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash_{\mathcal{M}}^\star (\lambda, q_{\texttt{accept}}, \rho), \text{ за някои } \lambda, \rho \in \Gamma^\star\}.$$

На англ. **semidecidable language**. В литературата се използва и названието **рекурсивно номеруем език**.

- 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 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. **decidable language**. В литературата се използва и названието **рекурсивен 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.

Твърдение 5.1. Ако L е разрешим език над азбуката Σ, то $\Sigma^\star \setminus L$ също е разрешим език.

Упътване. Просто разменяме $q_{\texttt{accept}}$ и $q_{\texttt{reject}}$. □

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 5.1. 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.

Вече сме срещали езикът L много пъти. Знаем много добре, 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.

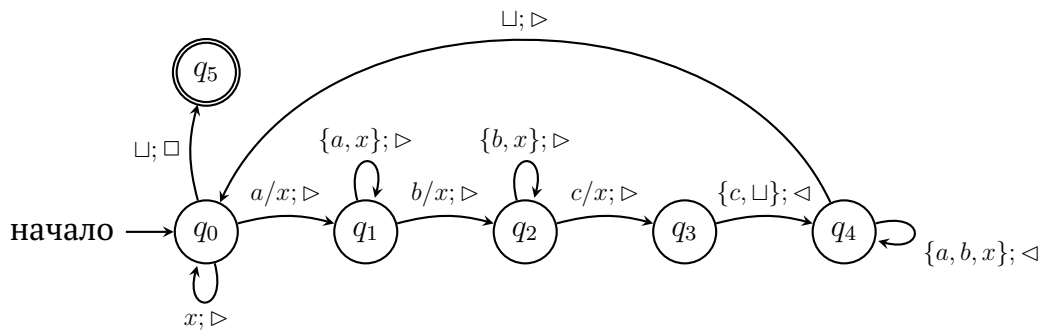
(2) Чете x-ове надясно по лентата докато срещне първото b и го замества с x. Отива на стъпка (3).

(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 $q_{\texttt{reject}}$.

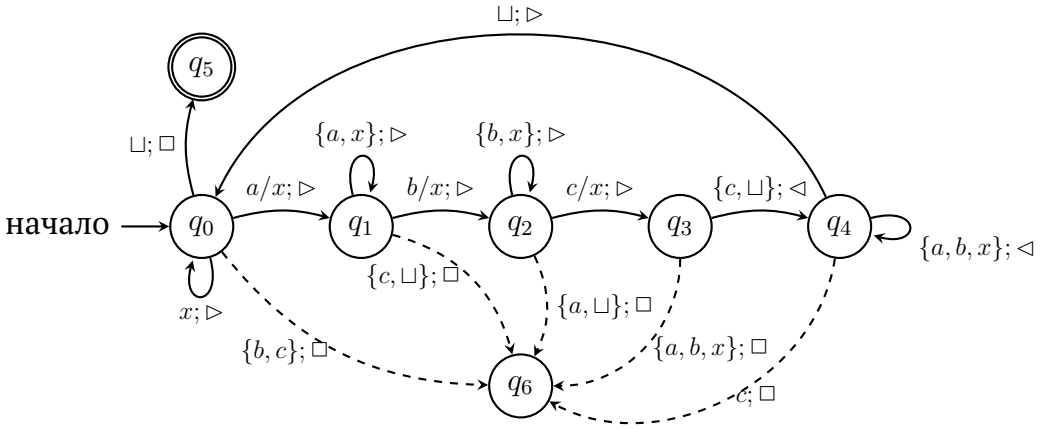


Фигура 5.4: Част от детерминистична машина на Тюринг $\mathcal{M}$, за която разрешава езика $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. Остава да се добави $q_{\texttt{reject}}$ и да се попълни δ функцията до тотална.

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 5.5: Детерминирана 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 [10, стр. 155] е дадено по-различно решение. Тук следваме [25, стр. 173]. Там има малка грешка.

Пример 5.2. 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). Това запаметяване става в състоянията на машината на Тюринг.

(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).

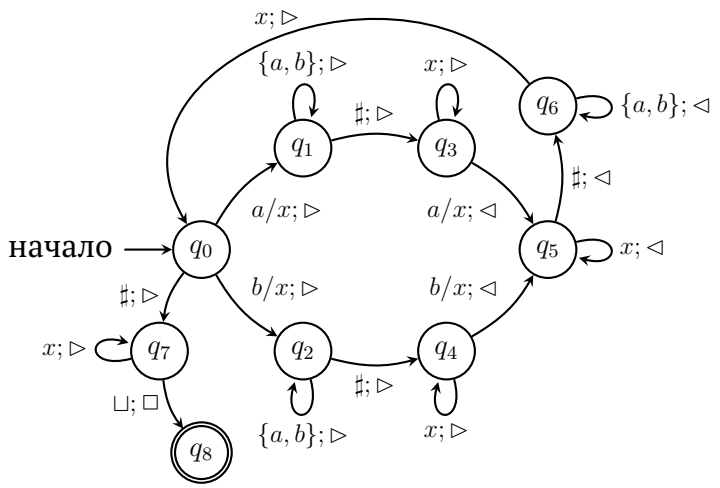
(6) Прочита $\sharp$ надясно по лентата и чете надясно x-ове докато не срещне $\sqcup$. Алгоритъмът завършва успешно.

Обърнете внимание, че този алгоритъм има времева сложност $\mathcal{O}(n^2)$. Ще видим в Пример 5.3, 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 5.6: 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 5.1. Докажете, че езикът $L = \{\, \omega \cdot \omega \mid \omega \in \{a,b\}^\star \,\}$ е разрешим.

5.1 Многолентови машини на Тюринг

Многолентови детерминистични машини на Тюринг

Детерминирана машина на Тюринг с k ленти има същата дефиниция като еднолентова машина на Тюринг с единствената разлика, че функцията на преходите δ приема следния вид:

$$\delta : Q' \times \Gamma^k \to Q \times \Gamma^k \times \{\triangleleft, \triangleright, \square\}^k,$$

където $Q' = Q \setminus \{q_{\texttt{accept}}, q_{\texttt{reject}}\}$. Това означава, че имаме k на брой ленти, всяка с отделна четяща глава, която се движи независимо от другите. Приемаме, че първоначално входната дума α е записана върху първата лента, а всички останали ленти са празни, т.е. запълнени са със символа $\sqcup$.

Тук означаваме $\vec{a} = (a_0, \ldots, a_{k-1})$.

За две n-орки от думи $\vec{\alpha}$ и $\vec{\beta}$, дефинираме n-орката

$$\vec{\alpha} \cdot \vec{\beta} \stackrel{\text{деф}}{=} (\alpha_1 \cdot \beta_1, \alpha_2 \cdot \beta_2, \ldots, \alpha_n \cdot \beta_n).$$

Сега дефинираме релацията $\vdash_{\mathcal{M}}$ над множеството $(\Gamma^\star)^k \times Q \times (\Gamma^+)^k$ по следния начин:

$$\frac{\delta(q, \vec{x}) = (q', \vec{y}, \vec{d}) \qquad \forall i < k : (\lambda_i, x_i\rho_i) \vdash_{y_i, d_i} (\lambda_i', \rho_i')}{(\vec{\lambda}, q, \vec{x} \cdot \vec{\rho}) \vdash_{\mathcal{M}} (\vec{\lambda}', q', \vec{\rho}')}$$

Фигура 5.7: Едностъпков преход в k-лентова детерминистична машина на Тюринг

Многолентови недетерминистични машини на Тюринг

Аналогично, тук разликата е в дефиницията на функцита Δ. Сега тя е следната:

$$\Delta : Q' \times \Gamma^k \to \mathcal{P}(Q \times \Gamma^k \times \{\triangleleft, \triangleright, \square\}^k),$$

където $Q' = Q \setminus \{q_{\texttt{accept}}, q_{\texttt{reject}}\}$.

$$\frac{\Delta(q, \vec{x}) \ni (q', \vec{y}, \vec{d}) \qquad \forall i < k : (\lambda_i, x_i\rho_i) \vdash_{y_i, d_i} (\lambda_i', \rho_i')}{(\vec{\lambda}, q, \vec{x} \cdot \vec{\rho}) \vdash_{\mathcal{N}} (\vec{\lambda}', q', \vec{\rho}')}$$

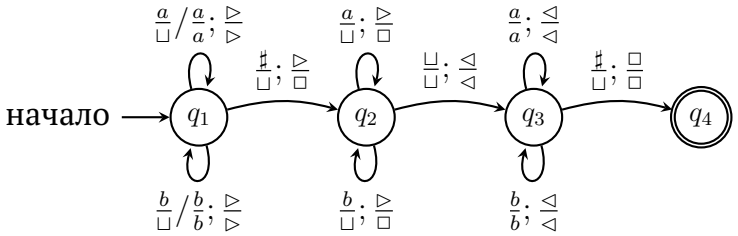
Фигура 5.8: Едностъпков преход в k-лентова недетерминистична машина на Тюринг

Примери

Пример 5.3. 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 $q_{\texttt{reject}} = q_5$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp, \sqcup\}$;
- $\delta : Q' \times \Gamma^2 \to Q \times \Gamma^2 \times \{\lhd, \rhd, \square\}^2$, където $Q' = \{q_1, q_2, q_3\}$.



Фигура 5.9: 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 5.2. За всяка k-лентова машина на Тюринг $\mathcal{M}$ съществува еднолентова машина на Тюринг $\mathcal{M}'$, такава че $\mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{M}')$.

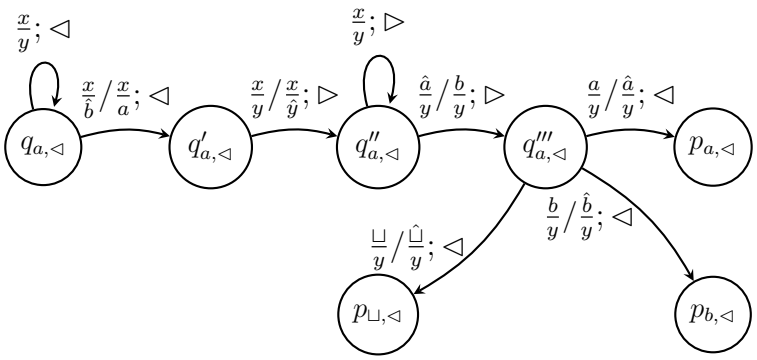
В [25, стр. 177] конструкцията е малко по-различна. Там съдържанието на всяка лента се поставя последователно върху една лента, като се разделят със специален символ. Тук следваме [10, стр. 162].

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

$$\frac{\hat{a}_1}{\hat{\sqcup}} \frac{a_2}{\sqcup} \cdots \frac{a_n}{\sqcup}$$

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 5.10: 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).
- 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.

✍ Помислете как да обобщите тази идея за машина на Тюринг с n ленти. Важното е, че във всяко състояние кодираме крайна информация.

- Във Фигура 5.10, $\frac{x}{y}$ 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.

5.2 Изчислими функции

Възможно е да се дефинира и за двулентова машина на Тюринг, като $f(\alpha)$ ще бъде върху втората лента.

Една *тотална* функция $f : \Sigma^\star \to \Sigma^\star$ се нарича изчислима с машина на Тюринг $\mathcal{M}$, ако за всяка дума $\alpha \in \Sigma^\star$,

$$(\varepsilon, q_{\texttt{start}}, \alpha\sqcup) \vdash^\star_{\mathcal{M}} (\sqcup^m, q_{\texttt{accept}}, f(\alpha)\sqcup^k), \text{ за някои } m, k \in \mathbb{N}.$$

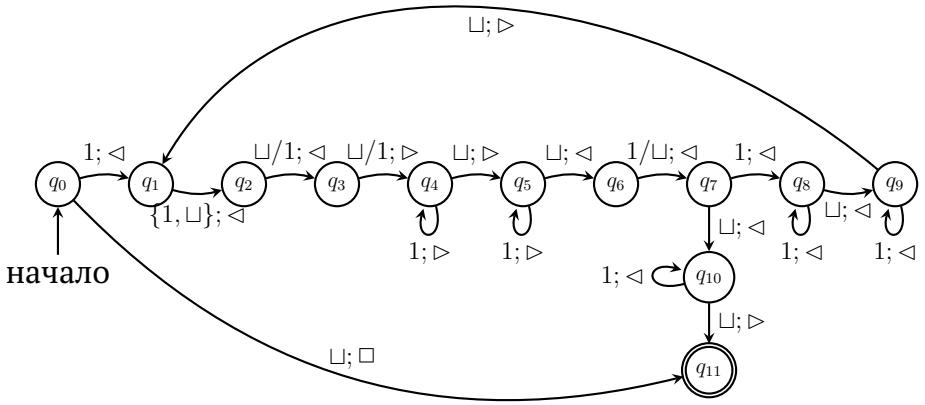
Това означава, че машината на Тюринг $\mathcal{M}$ винаги завършва. Лесно се съобразява, че езикът $\texttt{Graph}(f) = \{\, \alpha\sharp f(\alpha) \mid \alpha \in \Sigma^\star \,\}$ е разрешим.

Задача 5.2. 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 $f : \Sigma^\star \to \Sigma^\star$. □

При двулентова машина на Тюринг тази задача е много по-лесна и сложността ще бъде $\mathcal{O}(n)$ вместо $\mathcal{O}(n^2)$.

Пример 5.4. 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.



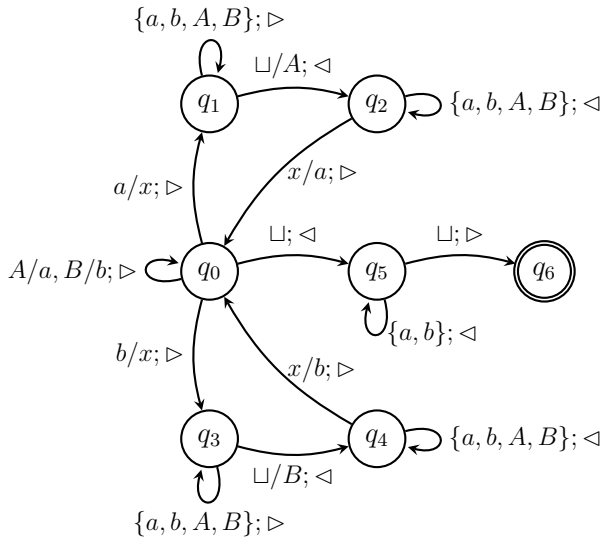
Фигура 5.11: Машина на Тюринг за функцията $f(1^n) = 1^{2n}$.

Това пак става много по-лесно с двулентова машина на Тюринг и сложността пак ще бъде $\mathcal{O}(n)$ вместо $\mathcal{O}(n^2)$.

Пример 5.5. 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.

- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{a, b, x, A, B\}$;

- $q_{\texttt{start}} \stackrel{\text{деф}}{=} q_0$ и $q_{\texttt{accept}} \stackrel{\text{деф}}{=} q_6$



Фигура 5.12: 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 $abab$.

$$\begin{aligned}(q_0, \underline{a}b\sqcup) &\vdash_{\mathcal{M}} (q_1, x\underline{b}\sqcup) \vdash_{\mathcal{M}} (q_1, xb\underline{\sqcup}) \vdash_{\mathcal{M}} (q_2, x\underline{b}A) \vdash_{\mathcal{M}} (q_2, \underline{x}bA)\\ &\vdash_{\mathcal{M}} (q_0, a\underline{b}A) \vdash_{\mathcal{M}} (q_3, ax\underline{A}) \vdash_{\mathcal{M}} (q_3, axA\underline{\sqcup}) \vdash_{\mathcal{M}} (q_4, ax\underline{A}B)\\ &\vdash_{\mathcal{M}} (q_4, a\underline{x}AB) \vdash_{\mathcal{M}} (q_0, ab\underline{A}B) \vdash_{\mathcal{M}} (q_0, aba\underline{B}) \vdash_{\mathcal{M}} (q_0, abab\underline{\sqcup})\\ &\vdash_{\mathcal{M}} (q_5, aba\underline{b}) \vdash_{\mathcal{M}} (q_5, ab\underline{a}b) \vdash_{\mathcal{M}} (q_5, a\underline{b}ab) \vdash_{\mathcal{M}} (q_5, \underline{a}bab)\\ &\vdash_{\mathcal{M}} (q_5, \sqcup abab) \vdash_{\mathcal{M}} (q_6, \underline{a}bab).\end{aligned}$$

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.

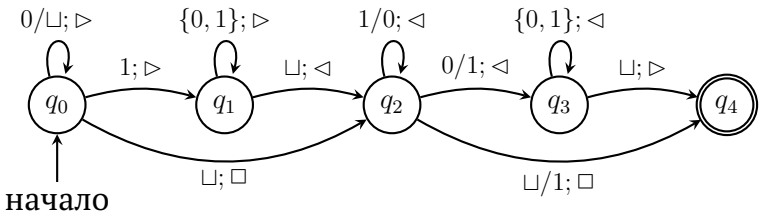
Пример 5.6. Да разгледаме тоталната функция $f : \{0,1\}^\star \to 1 \cdot \{0,1\}^\star$, където

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 $\overline{\alpha}_{(2)} + 1$.

$$\overline{f(\alpha)}_{(2)} \stackrel{\text{деф}}{=} \overline{\alpha}_{(2)} + 1.$$

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 5.13: Машина на Тюринг изчисляваща f, за която $\overline{f(\alpha)}_{(2)} = \overline{\alpha}_{(2)} + 1$.

Да проследим изчислението на $\mathcal{M}$ върху вход 01011.

$$\begin{aligned}(q_0, 0\underline{1}011\sqcup) &\vdash_{\mathcal{M}} (q_0, \sqcup\underline{1}011\sqcup) \vdash_{\mathcal{M}} (q_1, \sqcup 1\underline{0}11\sqcup) \vdash_{\mathcal{M}} (q_1, \sqcup 10\underline{1}1\sqcup)\\ &\vdash_{\mathcal{M}} (q_1, \sqcup 101\underline{1}\sqcup) \vdash_{\mathcal{M}} (q_1, \sqcup 1011\underline{\sqcup}) \vdash_{\mathcal{M}} (q_2, \sqcup 101\underline{1}\sqcup)\\ &\vdash_{\mathcal{M}} (q_2, \sqcup 10\underline{1}0\sqcup) \vdash_{\mathcal{M}} (q_2, \sqcup 1\underline{0}00\sqcup) \vdash_{\mathcal{M}} (q_3, \sqcup\underline{1}100\sqcup)\\ &\vdash_{\mathcal{M}} (q_3, \underline{\sqcup}1100\sqcup) \vdash_{\mathcal{M}} (q_4, \sqcup\underline{1}100\sqcup).\end{aligned}$$

Задача 5.3. 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.

5.3 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 $\vdash_{\mathcal{N}}$ над множеството $\Gamma^\star \times Q \times \Gamma^+$, която ще казва как моментното описание на машината $\mathcal{N}$ се променя при изпълнение на една стъпка.

$$\frac{\Delta(q,x) \ni (q', y, d) \qquad (\lambda, x\rho) \vdash_{y,d} (\lambda', \rho')}{(\lambda, q, x\rho) \vdash_{\mathcal{N}} (\lambda', q', \rho')}$$

Фигура 5.14: Преход в еднолентова недетерминирана машина на Тюринг $\mathcal{N}$

Сега за всяко естествено число ℓ, ще дефинираме релацията $\vdash^{\ell}_{\mathcal{N}}$, която ще казва, че от конфигурацията κ можем да достигнем до конфигурацията κ' за ℓ на брой стъпки.

Тази дефиниция на релацията $\vdash^{\ell}_{\mathcal{N}}$ вече се повтаря няколко пъти.

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

С $\vdash^{\star}_{\mathcal{N}}$ ще означаваме рефлексивното и транзитивно затваряне на релацията $\vdash_{\mathcal{N}}$ или с други думи,

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

Тогава за недетерминирана машина на Тюринг $\mathcal{N}$,

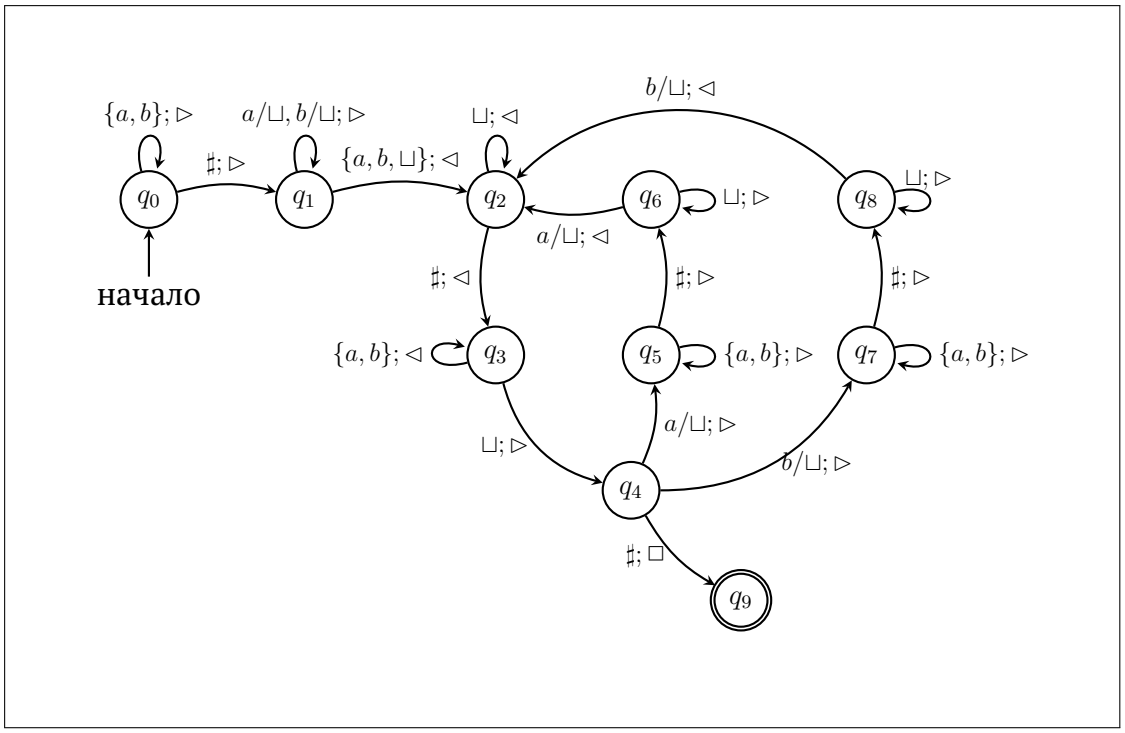
$$\mathcal{L}(\mathcal{N}) = \{\, \omega \in \Sigma^\star \mid (\varepsilon, q_{\texttt{start}}, \omega\sqcup) \vdash^{\star}_{\mathcal{N}} (\lambda, q_{\texttt{accept}}, \rho), \text{ за някои } \lambda, \rho \in \Gamma^\star\}.$$

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 $q_{\texttt{reject}}$.

Не е обяснено защо е разрешим.

Пример 5.7. 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.



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:

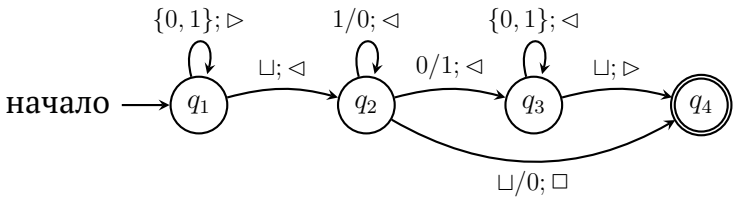
$$\varepsilon, 0, 1, \underbrace{00, 01, 10, 11}_{\text{от 0 до 3}}, \underbrace{000, 001, 010, 011, 100, 101, 110, 111}_{\text{от 0 до 7}}, 0000, 0001, \ldots$$

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 5.4. 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 еднолетновата детерминираната машина на Тюринг.

Упътване. Ако $\Sigma = \{0, 1\}$, то машината на Тюринг има следния вид:



Фигура 5.16: Генериране на следващата дума в каноничната наредба.

□

Теорема 5.1. Ако L се разпознава от *недетерминирана* машина на Тюринг $\mathcal{N}$, то L е разпознава и от *детерминирана* машина на Тюринг $\mathcal{D}$.

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

В [10, стр. 164] не е добре обяснено.

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 5.4* 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 5.4* 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 5.1. 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 $\mathcal{N}$.

□

5.4 Основни свойства

Твърдение 5.3. Ако езикът L е разрешим, то $\overline{L}$ също е разрешим език.

Означаваме $\overline{L} = \Sigma^\star \setminus L$. С други думи, твърдението ни казва, че разрешимите езици са затворени относно операцията допълнение. След малко в *Твърдение* **??** 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 5.4. Ако езиците L_1 и L_2 са разрешими , 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 5.5. 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 5.2 (Клини-Пост)**.** Езиците L и $\overline{L}$ са полуразрешими точно тогава, когато L е разрешим език.

✍ Дефинирайте сами новата машина на Тюринг $\mathcal{M}$.

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

Тук е удобно да индексираме от 1 вместо от 0.

Да приемем, че:

- $Q = \{q_1, q_2, \ldots, q_n\}$, 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 $x_s = \sqcup$;
- $d_1 = \square$, $d_2 = \triangleleft$, $d_3 = \triangleright$;

Ясно е, че тук съвсем спокойно можем да разгледаме и 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:

Ясно е, че не всяка дума над азбуката $\{0, 1\}$ е код на машина на Тюринг.

$$\ulcorner \mathcal{M} \urcorner \stackrel{\text{деф}}{=} 1110^\ell 11\ \texttt{code}_1\ 11\ \texttt{code}_2\ 11\ \cdots\ 11\ \texttt{code}_r\ 111.$$

По същия начин можем да дефинираме и код на краен автомат и стеков автомат.

Ще означаваме с $\mathcal{M}_\omega$ машината на Тюринг, чийто код е ω, т.е. искаме $\ulcorner \mathcal{M}_\omega \urcorner = \omega$. Важно свойство, което ще използваме често по-нататък, е че съществува алгоритъм, който при вход произволна дума $\omega \in \{0, 1\}^\star$, може да определи дали тази дума ω представлява код на машина на Тюринг.

Задача 5.5. Докажете, че езикът

$$L_{\texttt{code}} \stackrel{\text{деф}}{=} \{\ \omega \in \{0, 1\}^\star \mid \omega \text{ е код на машина на Тюринг}\ \}$$

е разрешим.

Диагоналният език

Теорема 5.3. 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.

Доказателство. Да допуснем, че $L_{\texttt{diag}}$ се разпознава от машина на Тюринг $\mathcal{M}$, т.е. $L_{\texttt{diag}} = \mathcal{L}(\mathcal{M})$. Тогава да видим какво имаме за думата $\ulcorner\mathcal{M}\urcorner$:

$$\ulcorner\mathcal{M}\urcorner \in L_{\texttt{diag}} \implies \ulcorner\mathcal{M}\urcorner \in \mathcal{L}(\mathcal{M}) \implies \ulcorner\mathcal{M}\urcorner \notin L_{\texttt{diag}},$$
$$\ulcorner\mathcal{M}\urcorner \notin L_{\texttt{diag}} \implies \ulcorner\mathcal{M}\urcorner \notin \mathcal{L}(\ulcorner\mathcal{M}\urcorner) \implies \ulcorner\mathcal{M}\urcorner \in L_{\texttt{diag}}.$$

Достигаме до противоречие. □

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. Тук е добре една безкрайна таблица да се нарисува.

Твърдение 5.6. 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 машина на Тюринг $\mathcal{M}'$, която, при вход произволна дума ω, работи по следния начин:

(1) $\mathcal{M}'$ проверява дали ω е код на машина на Тюринг $\mathcal{M}_\omega$.

(2) Ако ω е код на машина на Тюринг $\mathcal{M}_\omega$, то $\mathcal{M}'$ симулира работата на $\mathcal{M}_\omega$ върху ω.

(3) Ако след краен брой стъпки симулацията завърши с резултат, че $\mathcal{M}_\omega$ приема думата ω, то $\mathcal{M}'$ също завършва като приеме ω.

Това можем да го направим, защото знаем, че $L_{\texttt{code}}$ е разрешим.

Получаваме, че за всяка дума ω е изпълнена еквивалентността

$$\omega \in L_{\texttt{accept}} \;\Leftrightarrow\; \omega \in \mathcal{L}(\mathcal{M}'),$$

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 5.2. Съществува полуразрешим език L, за който $\overline{L}$ не е полуразрешим.

Упътване. Вземете езика $L = L_{\texttt{accept}}$. Знаем, че L е полуразрешим, но не е разрешим. Ако допуснем, че $\overline{L}$ е полуразрешим, то тогава от теоремата на Клини-Пост би следвало, че L е разрешим, което е противоречие. □

Твърдение 5.7. 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.

Упътване. Лесно се вижда, че $L_{\texttt{halt}}$ е полуразрешим. Да допуснем, че съществува машина на Тюринг $\mathcal{M}_{\texttt{halt}}$, която е разрешител за $L_{\texttt{halt}}$. Ще покажем, че тогава можем да построим разрешител $\mathcal{M}'$ за $L_{\texttt{accept}}$, което би било противоречие. Машината на Тюринг $\mathcal{M}'$ би работила така върху произволен вход ω:

(1) Ако ω не е код на машина на Тюринг, то $\mathcal{M}'$ директно отхвърляме ω.

(2) Ако ω е код на машина на Тюринг $\mathcal{M}_\omega$, то $\mathcal{M}'$ симулира работата на $\mathcal{M}_{\texttt{halt}}$ върху ω.

(3) Ако симулацията завърши с резултат, че $\mathcal{M}_{\texttt{halt}}$ отхвътрля думата ω, то $\mathcal{M}'$ завършва като отхвърля думата ω.

Щом $\omega \in L_{\texttt{halt}}$, то знаем, че рано или късно тази симулация ще завърши в някое състояние.

(4) Ако симулацията завърши с резултат, че $\mathcal{M}_{\texttt{halt}}$ приема думата ω, то $\mathcal{M}'$ симулира работата на $\mathcal{M}_\omega$ върху ω, докато тя завърши в състояние q.

- Ако $q = q_{\texttt{accept}}$ на $\mathcal{M}_\omega$, то $\mathcal{M}'$ завършва като приеме думата ω.
- В противен случай, $\mathcal{M}'$ завършва като отхвърля думата ω.

□

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 5.4. 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 5.3. 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.

5.5 Критерий за разрешимост

Сипсър нарича $\leq_m$ *mapping reducibility* [25, с. 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 5.4*, 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:

$$L_{\texttt{palin}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ съдържа само думи палиндроми}\}.$$

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 5.8. Езикът

$$L_{\Sigma^\star} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

не е разрешим.

$L_{\Sigma^\star}$ не е дори полуразрешим, но за момента не знаем как да докажем това.

Доказателство. 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 машина на Тюринг $\mathcal{M}_\omega$, то $f(\omega) \stackrel{\text{деф}}{=} \ulcorner \mathcal{M}' \urcorner$, където $\mathcal{M}'$, при вход произволна дума α, работи по следния начин:
  (1) Първоначално $\mathcal{M}'$ не обръща внимание на α, а $\mathcal{M}'$ симулира работата на $\mathcal{M}_\omega$ върху думата ω.
  (2) Ако след краен брой стъпки симулацията завърши с резултат, че $\mathcal{M}_\omega$ приема думата ω, то $\mathcal{M}'$ завършва като приема думата α.

Получаваме, че:

$$\mathcal{L}(\mathcal{M}') = \begin{cases} \Sigma^\star, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ \emptyset, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

Можем да заключим, че за произволна дума ω са изпълнени импликациите:

$$\begin{aligned} \omega \in L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \Sigma^\star \implies f(\omega) \in L_{\Sigma^\star}, \\ \omega \in L_{\texttt{code}} \setminus L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \emptyset \implies f(\omega) \notin L_{\Sigma^\star} \\ \omega \notin L_{\texttt{code}} &\implies f(\omega) = \omega \implies f(\omega) \notin L_{\Sigma^\star}, \end{aligned}$$

които можем да обобщим в следната еквивалентност:

$$\omega \in L_{\texttt{accept}} \Leftrightarrow f(\omega) \in L_{\Sigma^\star}$$

Ако допуснем, че $L_{\Sigma^\star}$ е разрешим език, то $L_{\texttt{accept}}$ също ще е разрешим, което е противоречие. □

Следствие 5.4. Езикът

$$\overline{L}_{\texttt{empty}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \neq \emptyset\}$$

е полуразрешим, но не е разрешим.

Упътване. Съобразете, че в може да използвате функцията f от доказателството на *Твърдение* **??** за да получите, че:

$$\omega \in L_{\texttt{accept}} \Leftrightarrow f(\omega) \in \overline{L}_{\texttt{empty}}.$$

□

Следствие 5.5. Езикът

$$L_{\texttt{empty}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

не е полуразрешим.

Упътване. Ако $L_{\texttt{empty}}$ беше разрешим, то неговото допълнение

$$\overline{L}_{\texttt{empty}} = L_{\texttt{code}} \setminus L_{\texttt{empty}}$$

щеше да е разрешим език, което е противоречие.

Ако $L_{\texttt{empty}}$ беше полуразрешим, тогава, използвайки, че $\overline{L}_{\texttt{empty}}$ е полуразрешим, от теоремата на Клини-Пост щеше да следва, че $L_{\texttt{empty}}$ е разрешим, което е противоречие □

Твърдение 5.9. Езикът

$L_{\texttt{reg}} \stackrel{\text{деф}}{=} \{\, \omega \mid \omega$ е код на машина на Тюринг и $\mathcal{L}(\mathcal{M}_\omega)$ е регулярен език $\}$

не е разрешим.

[25, стр. 219]

Доказателство. Да фиксираме един език, за който знаем, че не е регулярен, например, $\{0^n1^n \mid n \in \mathbb{N}\}$. 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 $\mathcal{M}'$, при вход произволна дума α, работи така:
  (1) Ако $\alpha = 0^n1^n$, за някое n, то $\mathcal{M}'$ приема думата α.
  (2) Ако α не е от вида 0^n1^n, тогава $\mathcal{M}'$ симулира работата на $\mathcal{M}_\omega$ върху думата ω.
  (3) Ако след краен брой стъпки симулацията завърши с резултат, че $\mathcal{M}_\omega$ приема думата ω, то $\mathcal{M}'$ завършва като приема α.

Използваме наготово, че $\{0,1\}^\star$ е регулярен език.

Получаваме, че:

$$\mathcal{L}(\mathcal{M}') = \begin{cases} \{0,1\}^\star, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ \{0^n1^n \mid n \in \mathbb{N}\}, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

Сега можем да заключим, че за произволна дума ω са изпълнени импликациите:

$$\begin{aligned} \omega \in L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \{0,1\}^\star \implies f(\omega) \in L_{\texttt{reg}}, \\ \omega \in L_{\texttt{code}} \setminus L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \{0^n1^n \mid n \in \mathbb{N}\} \implies f(\omega) \notin L_{\texttt{reg}}, \\ \omega \notin L_{\texttt{code}} &\implies f(\omega) = \omega \implies f(\omega) \notin L_{\texttt{reg}}, \end{aligned}$$

което можем да обединим в еквивалентността:

$$\omega \in L_{\texttt{accept}} \Leftrightarrow f(\omega) \in L_{\texttt{reg}}$$

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. Ще казваме, че $\mathscr{S}$ е свойство на полуразрешимите езици. Например,

$$\mathscr{S} = \{L \subseteq \Sigma^\star \mid L \text{ е регулярен език}\}.$$

$\mathscr{S}$ 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.

Задача 5.6. Докажете, че езикът

$$L_{\texttt{Dec}} = \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е разрешим}\}$$

не е разрешим.

Задача 5.7. Докажете, че езикът

$$L_{\texttt{palin}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ съдържа само думи палиндроми}\}.$$

не е разрешим.

Сега вече имаме достатъчно опит за да видим точно кои проблеми са разрешими.

Теорема 5.5 (Райс 1953 [22])**.** За всяко нетривиално свойство $\mathscr{S}$ на полуразрешимите езици, $\texttt{Code}(\mathscr{S})$ е неразрешим.

[10, стр. 188]

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 $\mathcal{L}(\mathcal{M}_L) = L$. Ще дефинираме тотална изчислима функция f, която при вход думата $\omega \in \{0,1\}^\star$ работи по следния начин:

Цел: да сведем ефективно $L_{\texttt{accept}}$ към $L_{\mathscr{S}}$

- 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 $\mathcal{M}'$, при вход произволна дума α, работи така:
  (1) първоначално $\mathcal{M}'$ не обръща внимание на входната дума α, а започва да симулира работата на $\mathcal{M}_\omega$ върху ω.
  (2) ако след краен брой стъпки симулацията завърши с резултат, че $\mathcal{M}_\omega$ приема думата ω, то $\mathcal{M}'$ започва да симулира работата на $\mathcal{M}_L$ върху входната дума α;
  (3) ако след краен брой стъпки симулацията завърши с резултат, че $\mathcal{M}_L$ приема думата α, то $\mathcal{M}'$ приема входната дума α;

Така получаваме, че:

$$\mathcal{L}(\mathcal{M}') = \begin{cases} L, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ \emptyset, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

Оттук заключаваме, че за произволна дума ω са изпълнени импликациите:

$$\begin{aligned} \omega \in L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L \implies f(\omega) \in \texttt{Code}(\mathscr{S}), \\ \omega \in L_{\texttt{code}} \setminus L_{\texttt{accept}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \emptyset \implies f(\omega) \notin \texttt{Code}(\mathscr{S}), \\ \omega \notin L_{\texttt{code}} &\implies f(\omega) = \omega \implies f(\omega) \notin \texttt{Code}(\mathscr{S}), \end{aligned}$$

които можем да обобщим в следната еквивалентност:

$$\omega \in L_{\texttt{accept}} \Leftrightarrow f(\omega) \in \texttt{Code}(\mathscr{S}).$$

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 5.6. 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

$$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

не е разрешим;

б) $\mathscr{S}$ е свойството за пълнота, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

не е разрешим;

в) $\mathscr{S}$ е свойството крайност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } |\mathcal{L}(\mathcal{M}_\omega)| < \infty\}$$

не е разрешим;

г) $\mathscr{S}$ е свойството безкрайност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } |\mathcal{L}(\mathcal{M}_\omega)| = \infty\}$$

не е разрешим;

д) $\mathscr{S}$ е свойството регулярност, т.е. езикът

$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е регулярен език}\}$

не е разрешим;

е) $\mathscr{S}$ е свойството безконтекстност, т.е. езикът

$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е безконтекстен}\}$

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.

ж) $\mathscr{S}$ е свойството разрешимост, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е разрешим}\}$$

не е разрешим.

Тук също - вече сме разгледали примери за полуразрешими езици, които не са разрешими.

5.6 Критерии за полуразрешимост

Вече знаем, че на практика всички интересни въпроси за полуразрешими езици не са разрешими. Сега да видим какво можем да кажем, ако се ограничим само до позитивната част на тези въпроси.

Лема 5.1. Нека $\mathscr{S}$ е свойство на полуразрешимите езици. Ако съществува безкраен език $L_0 \in \mathscr{S}$, който няма краен подезик 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 имаме следната връзка:

$$L_{\texttt{diag}} \leq_m \texttt{Code}(\mathscr{S}).$$

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 $\mathcal{M}'$, при вход произволна дума α, работи така:
  (1) Първоначално $\mathcal{M}'$ не обръща внимание на α, а симулира работата на $\mathcal{M}_\omega$ върху думата ω.
  (2) Ако симулацията завърши за по-малко от $|\alpha|$ на брой стъпки с резултат, че $\mathcal{M}_\omega$ приема ω, то $\mathcal{M}'$ завършва веднага като *отхвърля* α.
  (3) В противен случай, $\mathcal{M}'$ симулира работата на $\mathcal{M}_0$ върху α.
  (4) Ако симулацията завърши с резултат, че $\mathcal{M}_0$ приема α, то $\mathcal{M}'$ завършва като приеме думата α.

Така получаваме, че:

$$\mathcal{L}(\mathcal{M}') = \begin{cases} \{\alpha \in L_0 \mid |\alpha| < \texttt{step}_\omega\}, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ L_0, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega), \end{cases}$$

където $\texttt{step}_\omega$ е минималният брой стъпки необходими на $\mathcal{M}_\omega$ за да приеме думата ω. Заключаваме, че за произволна дума $\omega \in \{0,1\}^\star$ са изпълнени импликациите:

$$\begin{aligned} \omega \in L_{\texttt{diag}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L_0 \implies f(\omega) \in \texttt{Code}(\mathscr{S}) \\ \omega \in L_{\texttt{code}} \setminus L_{\texttt{diag}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) \text{ е краен подезик на } L_0 \implies f(\omega) \notin \texttt{Code}(\mathscr{S}) \\ \omega \notin L_{\texttt{code}} &\implies f(\omega) = \omega \implies f(\omega) \notin \texttt{Code}(\mathscr{S}). \end{aligned}$$

които можем да обединим в следната еквивалентност:

$$\omega \in L_{\mathtt{diag}} \Leftrightarrow f(\omega) \in \mathtt{Code}(\mathscr{S}),$$

Оттук следва, че $\mathtt{Code}(\mathscr{S})$ не е полуразрешим, защото от *Теорема 5.3* ние знаем, че $L_{\mathtt{diag}}$ не е полуразрешим. □

Следствие 5.7. Директно от *Лема 5.1* следва, че за всяко от следните свойства $\mathscr{S}$ на полуразрешимите езици, $\mathtt{Code}(\mathscr{S})$ **не** е полуразрешим език, където:

Защо не можем да използваме *Лема 5.1* за да докажем, че свойството празнота не е полуразрешимо, както и свойството регулярност, разрешимост, полуразрешимост?

- $\mathscr{S}$ е свойството безкрайност, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } |\mathcal{L}(\mathcal{M}_\omega)| = \infty\}$$

  не е полуразрешим;
- $\mathscr{S}$ е свойството за пълнота, т.е. езикът

  $$\mathtt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

  не е полуразрешим;
- $\mathscr{S}$ е свойството неразрешимост, т.е. езикът

  $\mathtt{Code}(\mathscr{S}) = \{\omega \mid \omega$ е код на машина на Тюринг и $\mathcal{L}(\mathcal{M}_\omega)$ не е разрешим$\}$

  не е полуразрешим;
- $\mathscr{S}$ е свойството неполуразрешимост, т.е. езикът

  $\mathtt{Code}(\mathscr{S}) = \{\omega \mid \omega$ е код на машина на Тюринг и $\mathcal{L}(\mathcal{M}_\omega)$ не е полуразрешим$\}$

  не е полуразрешим;
- $\mathscr{S}$ е свойството нерегулярност, т.е. езикът

  $\mathtt{Code}(\mathscr{S}) = \{\omega \mid \omega$ е код на машина на Тюринг и $\mathcal{L}(\mathcal{M}_\omega)$ не е регулярен$\}$

  не е полуразрешим.

Лема 5.2. Нека L_1 е език в $\mathscr{S}$ и нека L_2 е полуразрешим език, като $L_1 \subsetneq L_2$ и $L_2 \notin \mathscr{S}$. Тогава $\mathtt{Code}(\mathscr{S})$ не е полуразрешим език.

Упътване. Нека $L_1 = \mathcal{L}(\mathcal{M}_1)$ и $L_2 = \mathcal{L}(\mathcal{M}_2)$. Ще докажем, че

$$L_{\mathtt{diag}} \leq_m \mathtt{Code}(\mathscr{S}).$$

Това означава, че за полуразрешим език $\mathtt{Code}(\mathscr{S})$, ако $L_1 \in \mathscr{S}$ и $L_1 \subseteq L_2$, като L_2 е полуразрешим, то $L_2 \in \mathscr{S}$.

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 $\mathcal{M}_\omega$, тогава $f(\omega) \stackrel{\text{деф}}{=} \ulcorner \mathcal{M}' \urcorner$, където $\mathcal{M}'$, при вход произволна дума α, работи така:

  (1) $\mathcal{M}'$ симулира едновременно две изчисления - работата на $\mathcal{M}_1$ върху α и работата на $\mathcal{M}_\omega$ върху ω, докато намери стъпка s, такава че:

  (2) ако симулацията на $\mathcal{M}_1$ завършва за s на брой стъпки като приема думата α, то $\mathcal{M}'$ завършва като приема думата α;

  (3) ако симулацията на $\mathcal{M}_\omega$ завършва за s на брой стъпки като приема думата ω, то $\mathcal{M}'$ продължава като симулира работата $\mathcal{M}_2$ върху α.

  (4) Ако симулацията на $\mathcal{M}_2$ завърши като приема думата α, то $\mathcal{M}'$ завършва като приема думата α.

Съобразете сами, че получаваме следното:

$$\mathcal{L}(\mathcal{M}') = \begin{cases} L_2, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ L_1, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

Понеже $L_2 \notin \mathscr{S}$, а $L_1 \in \mathscr{S}$, можем да заключим, че за произволна дума $\omega \in \{0,1\}^\star$ са изпълнени импликациите:

$$\begin{aligned} \omega \in L_{\texttt{diag}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L_1 \implies f(\omega) \in \texttt{Code}(\mathscr{S}) \\ \omega \in L_{\texttt{code}} \setminus L_{\texttt{diag}} &\implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L_2 \implies f(\omega) \notin \texttt{Code}(\mathscr{S}) \\ \omega \notin L_{\texttt{code}} &\implies f(\omega) = \omega \implies f(\omega) \notin \texttt{Code}(\mathscr{S}), \end{aligned}$$

които можем да обединим в следната еквивалентност:

$$\omega \in L_{\texttt{diag}} \Leftrightarrow f(\omega) \in \texttt{Code}(\mathscr{S}),$$

Това означава, че ефективно можем да сведем въпрос за принадлежност в $L_{\texttt{diag}}$ към въпрос за принадлежност в $\texttt{Code}(\mathscr{S})$. Следователно, ако $\texttt{Code}(\mathscr{S})$ е полуразрешим език, то $L_{\texttt{diag}}$ е полуразрешим език, което е противоречие. □

Следствие 5.8. Директно от *Лема 5.2* следва, че за всяко от следните свойства $\mathscr{S}$ на полуразрешимите езици, $\texttt{Code}(\mathscr{S})$ **не** е полуразрешим език, където:

- $\mathscr{S}$ е свойството крайност, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } |\mathcal{L}(\mathcal{M}_\omega)| < \infty\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството празнота, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството разрешимост, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е разрешим}\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството безконтекстност, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е безконтекстен}\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството регулярност, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \mid \omega \text{ е код на машина на Тюринг и } \mathcal{L}(\mathcal{M}_\omega) \text{ е регулярен}\}$$

  не е полуразрешим.

Теорема 5.6 (Райс-Шапиро). Нека $\texttt{Code}(\mathscr{S})$ е полуразрешим език. Тогава е изпълнена еквивалентността:

$$L \in \mathscr{S} \;\Leftrightarrow\; (\exists L_0 \subseteq \Sigma^\star)[L_0 \text{ е краен и } L_0 \subseteq L \;\&\; L_0 \in \mathscr{S}].$$

Доказателство. *Лема 5.1* може да се формулира така: ако $\texttt{Code}(\mathscr{S})$ е полуразрешим, то всеки език $L \in \mathscr{S}$ има краен подезик $L_0 \subseteq L$, за който $L_0 \in \mathscr{S}$. Това ни дава посоката $(\Rightarrow)$ на *Теорема 5.6*.

Лема 5.2 може да се формулира така: ако $\texttt{Code}(\mathscr{S})$ е полуразрешим, то за всеки два езика $L_1 \subseteq L_2$, ако $L_1 \in \mathscr{S}$, то $L_2 \in \mathscr{S}$. Това ни дава посоката $(\Leftarrow)$ на *Теорема 5.6*. □

5.7 Проблемът за съответствието на Пост

На англ. *Post's correspondence problem* [9, стр. 392], но по-добре е обяснено в [25, стр. 227]. Сипсър нарича двойките домино.

Пример за проблема за съответствието на Пост се нарича всяка крайна редица от елементи на $\Sigma^+ \times \Sigma^+$, които ние ще означаваме така:

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_n\\ \beta_n\end{bmatrix}.$$

✍ Съобразете, че задачата е тривиална, ако:

- ако $|\alpha_i| = |\beta_i|$ за всяко i, или
- ако $|\Sigma| = 1$.

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:

$$\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_n} = \beta_{i_1}\beta_{i_2}\cdots\beta_{i_n}.$$

Задача 5.8. Намерете решение на следния пример за PCP:

$$\begin{bmatrix}ab\\ abab\end{bmatrix}, \begin{bmatrix}b\\ a\end{bmatrix}, \begin{bmatrix}aba\\ b\end{bmatrix}, \begin{bmatrix}aa\\ a\end{bmatrix}.$$

Взел съм този пример от [25, стр. 239].

Решение.

$$\begin{bmatrix}ab\cdot ab\cdot aba\cdot b\cdot b\cdot aa\cdot aa\\ abab\cdot abab\cdot b\cdot a\cdot a\cdot a\cdot a\end{bmatrix}$$

□

Понеже един пример P за съответствието на Пост представлява крайна редица от думи над дадена азбука Σ, нека $\ulcorner P \urcorner$ е дума над азбуката $\Sigma \cup \{\sharp\}$, която кодира P по следния начин:

$$\ulcorner P \urcorner \stackrel{\text{деф}}{=} \alpha_1\sharp\beta_1\sharp\alpha_2\sharp\beta_2\sharp\cdots\sharp\alpha_n\sharp\beta_n.$$

Нека положим

$$\texttt{PCP} \stackrel{\text{деф}}{=} \{\ulcorner P \urcorner \mid P \text{ е пример с решение за ПСП}\}.$$

Модифициран проблем за съответствието на Пост (МПСП)

Тук искаме винаги да започваме с първата двойка.

Казваме, че MPCP има решение, ако съществува произволна редица от индекси $i_1, \ldots, i_n$ (може и празна), такава че:

$$\alpha_1\alpha_{i_1}\cdots\alpha_{i_n} = \beta_1\beta_{i_1}\cdots\beta_{i_n}.$$

Нека положим

$$\texttt{MPCP} \stackrel{\text{деф}}{=} \{\ulcorner P \urcorner \mid P \text{ е пример с решение за МПСП}\}.$$

Лема 5.3. Съществува алгоритъм, който свежда MPCP към PCP.

Упътване. Нека имаме пример за MPCP :

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix}.$$

Да фиксираме символи $\star, \$$, които не са от Σ. За произволна дума $\alpha = a_1 \cdots a_n$ 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 5.9. 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 5.7 (Емил Пост [20])**.** 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

$$L_{\texttt{accept}} \leq_m \texttt{MPCP}.$$

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

$$\omega \in L_{\texttt{accept}} \;\Leftrightarrow\; f(\omega) \in \texttt{MPCP}.$$

Сега неформално ще опишем работата на функцията f. Нека фиксираме символа $\sharp \notin \Gamma$.

1) Нека имаме като вход дума ω, която е код на машина на Тюринг $\mathcal{M}_\omega$.

Горната част на доминото се опитва да настигне долната част.

2) Започваме като добавяме за думата $\omega = a_1 \cdots a_n$ 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 $\begin{bmatrix} \sharp \\ \sharp \end{bmatrix}$.

Когато достигнем до приемащо състояние, то започваме да трием съдържанието на доминото за да можем да изравним двете части на доминото.

9) За всеки $x \in \Gamma$, добавяме двойката $\begin{bmatrix} x q_{\texttt{accept}} \\ q_{\texttt{accept}} \end{bmatrix}$ и $\begin{bmatrix} q_{\texttt{accept}} x \\ q_{\texttt{accept}} \end{bmatrix}$.

10) За да завършим, добавяме двойката $\begin{bmatrix} q_{\texttt{accept}} \sharp\sharp \\ \sharp \end{bmatrix}$.

□

В практиката обикновено се интересуваме от еднозначни безконтекстни граматики. За съжаление, не съществува алгоритъм, който да ни каже дали дадена безконтекстна граматика е еднозначна или не.

Вече би трябвало да ви е очевидно защо AMBIG е полуразрешим език. Нали?

Следствие 5.10. Следният език

$$\texttt{AMBIG} \stackrel{\text{деф}}{=} \{\ulcorner G \urcorner \mid G \text{ е неоднозначна безконтекстна граматика}\}.$$

е полуразрешим, но не е разрешим език.

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 5.11. Следният език

$$\texttt{INTERSECT} \stackrel{\text{деф}}{=} \{\ulcorner G_1 \urcorner \sharp \ulcorner G_2 \urcorner \mid \mathcal{L}(G_1) \cap \mathcal{L}(G_2) \neq \emptyset\}$$

е полуразрешим, но не е неразрешим.

Упътване. Да разгледаме един пример за PCP над азбуката Σ:

$$\left[\begin{matrix}\alpha_1\\ \beta_1\end{matrix}\right], \left[\begin{matrix}\alpha_2\\ \beta_2\end{matrix}\right], \ldots, \left[\begin{matrix}\alpha_n\\ \beta_n\end{matrix}\right].$$

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 5.8*.

По него можем ефективно да построим следните две безконтекстни граматики 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

$$\texttt{PCP} \leq_m \texttt{INTERSECT}.$$

□

5.8 Историята от всички изчисления

[10, 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 5.10. 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 $\vdash_{\mathcal{M}}$:

$$(\alpha_1, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1 y, p, \alpha_2) \quad // \text{ ако } q \xrightarrow{x/y;\triangleright} p$$
$$(\alpha_1, q, \varepsilon) \vdash_{\mathcal{M}} (\alpha_1 y, p, \varepsilon) \quad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleright} p$$
$$(\alpha_1 z, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1, p, zy\alpha_2) \quad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$(\alpha_1 z, q, \varepsilon) \vdash_{\mathcal{M}} (\alpha_1, p, z) \quad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleleft} p$$
$$(\varepsilon, q, x\alpha_2) \vdash_{\mathcal{M}} (\varepsilon, p, \sqcup y\alpha_2) \quad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$(\alpha_1, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1, p, y\alpha_2) \quad // \text{ ако } q \xrightarrow{x/y;\square} p.$$

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.

$$\alpha_1 q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} p y \alpha_1^{\mathtt{rev}} \quad // \text{ ако } q \xrightarrow{x/y;\triangleright} p$$
$$\alpha_1 q \sharp p y \alpha_1^{\mathtt{rev}} \quad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleright} p$$
$$\alpha_1 z q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} y z p \alpha_1^{\mathtt{rev}} \quad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$\alpha_1 z q \sharp y z p \alpha_1^{\mathtt{rev}} \quad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleleft} p$$
$$\alpha_1 q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} y p \alpha_1^{\mathtt{rev}} \quad // \text{ ако } q \xrightarrow{x/y;\square} p$$
$$q x \alpha_2 \sharp \alpha_2^{\mathtt{rev}} y \sqcup p \quad // \text{ ако } q \xrightarrow{x/y;\triangleleft} p$$
$$q \sharp y \sqcup p \quad // \text{ ако } q \xrightarrow{\sqcup/y;\triangleleft} p$$

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 $\texttt{History}(\mathcal{M})$.

Лема 5.4. За всяка машина на Тюринг $\mathcal{M}$ е изпълнено, че: [10, стр. 201]

$$\texttt{History}(\mathcal{M}) = L_1 \cap L_2,$$

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 5.5. 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 5.9. Обяснете как може ефективно да се кодира всяка безконтекстна граматика G като дума $\ulcorner G \urcorner$ над азбуката $\{0, 1\}$.

Теорема 5.8. 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.

Да напомним, че от *Следствие 5.11* знаем, че INTERSECT не е разрешим, но 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

$$f(\ulcorner \mathcal{M} \urcorner) = \ulcorner G_1 \urcorner \cdot \ulcorner G_2 \urcorner.$$

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 5.6. 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:

Можем да опишем това свойство с регулярен език

- 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
  - $\omega_i \not\vdash_{\mathcal{M}} \omega_{i+1}^{\texttt{rev}}$, т.е. $\omega_i \sharp \omega_{i+1}^{\texttt{rev}} \in \texttt{Invalid}(\mathcal{M})$, за някое нечетно $i < n$, или
  - $\omega_i^{\texttt{rev}} \not\vdash_{\mathcal{M}} \omega_{i+1}$, т.е. $\omega_i^{\texttt{rev}} \sharp \omega_{i+1} \in \texttt{Invalid'}(\mathcal{M})$, 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 $\texttt{Invalid}(\mathcal{M})$ и $\texttt{Invalid'}(\mathcal{M})$. □

Теорема 5.9. За дадена азбука Σ, езикът

$$\mathtt{All}_{\mathtt{CFG}} \overset{\text{деф}}{=} \{\ulcorner G \urcorner \mid G \text{ е безконтекстна граматика и } \mathcal{L}(G) = \Sigma^\star\}$$

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 5.12. 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 5.10 (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:

[10, 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.

Съществуват езици от $\mathcal{C}$, които не притежават свойството P и такива, които го притежават.

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

$$L/_a = \{\omega \mid \omega a \in L\}.$$

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:

$$L = \Sigma^\star \ \Leftrightarrow\ \ulcorner \hat{L} \urcorner \in P. \tag{5.1}$$

- 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 $\ulcorner L \urcorner \in P$, то за $L/_\beta \stackrel{\text{деф}}{=} \{\alpha \mid \alpha\beta \in L\}$ е изпълнено P.

- 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.

От (5.1) следва, че P е разрешимо свойство точно тогава, когато въпросът "$= \Sigma^\star$" за езиците от $\mathcal{C}$ е разрешим, което е противоречие. □

Следствие 5.13. 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 5.9*, че $= \Sigma^\star$ за безконтекстни граматики е неразрешим за достатъчно голяма азбука Σ. Тогава от теоремата на Грейбах следва, че Reg е неразрешим език. □

5.9 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 5.10. Докажете, че класът $\mathcal{P}$ е затворен относно допълнение, обединение, сечение и конкатенация.

Задача 5.11. Докажете, че класът $\mathcal{P}$ е затворен относно операцията звезда на Клини.

Теорема 5.11. $\mathcal{NP} \subseteq \mathcal{EXP}$.

Твърдение 5.11. 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. □

5.10 Задачи

Задача 5.12. 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 5.13. 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 5.14. 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 5.15. 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

[1] Alfred Aho и др. *Compilers: principles, techniques and tools*. 2-е изд. Pearson Education, 2007.

[2] D. N. Arden. “Delayed-logic and finite-state machines”. В: *2nd Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1961)*. 1961, с. 133—151.

[3] Y. Bar-Hillel, M. Perles и E. Shamir. “On formal properties of simple phrase structure grammars”. В: *STUF - Language Typology and Universals* 14.1-4 (1961), с. 143—172. doi: 10.1524/stuf.1961.14.14.143.

[4] Jean Berstel. “Context-Free Languages”. В: *Transductions and Context-Free Languages*. Wiesbaden: Vieweg+Teubner Verlag, 1979, с. 22—50. isbn: 978-3-663-09367-1. doi: 10.1007/978-3-663-09367-1_2. url: https://doi.org/10.1007/978-3-663-09367-1_2.

[5] Janusz A. Brzozowski. “Derivatives of Regular Expressions”. В: *Journal of the ACM* 11.4 (окт. 1964), с. 481—494. issn: 0004-5411. doi: 10.1145/321239.321249.

[6] John H. Conway. *Regular algebra and finite machines*. Chapman и Hall, 1971. isbn: 0412106205.

[7] Ding-Zhu Du и Ker-I. Ko. *Problem Solving in Automata, Languages, and Complexity*. OCLC: 475923550. John Wiley & Sons, 2004. isbn: 978-0-471-46408-2.

[8] Javier Esparza. *Automata theory. An algorithmic approach*. 2017. url: https://www7.in.tum.de/~esparza/automatanotes.html.

[9] John E. Hopcroft, Rajeev Motwani и Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. second. Addison-Wesley, 2001.

[10] John E. Hopcroft и Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. first. Addison-Wesley, 1979.

[11] Bakhadyr Khoussainov и Anil Nerode. *Automata Theory and its Applications*. Springer, 2001.

[12] S. C. Kleene. “Representation of events in nerve nets and finite automata”. В: *Automata Studies*. Под ред. на C. E. Shannon и J. McCarthy. Princeton University Press, 1956, с. 3—42. doi: 10.1515/9781400882618-002.

[13] Dexter Kozen. *Automata and Computability*. Springer, 1997.

[14] Anil Maheshwari и Michael Smid. *Introduction to Theory of Computation*. 2019. url: https://cglab.ca/~michiel/TheoryOfComputation/.

[15] J. Myhill. “Finite automata and the representation of events”. В: *WADD TR-57-624, Wright Patterson AFB, Ohio* (1957), с. 112—137.

[16] A. Nerode. “Linear automata transformation”. В: *Proceedings of AMS* 9 (1958), с. 541—544.

[17] Anil Nerode и Richard Shore. *Logic for Applications*. Springer-Verlag, 1993. doi: 10.1007/978-1-4612-0649-1.

[18] Christos Papadimitriou и Harry Lewis. *Elements of the Theory of Computation*. Prentice-Hall, 1998.

[19] Alberto Pettorossi. *Automata Theory and Formal Languages: Fundamental Notions, Theorems, and Techniques*. Undergraduate Topics in Computer Science. Springer International Publishing, 2022. doi: 10.1007/978-3-031-11965-1.

[20] Emil L. Post. “A variant of a recursively unsolvable problem”. В: *Bull. Amer. Math. Soc.* 52 (1946), с. 264—268. doi: https://doi.org/10.1090/S0002-9904-1946-08555-9.

[21] M. O. Rabin и D. Scott. “Finite automata and their decision problems”. В: *IBM Journal of Research and Development* 3 (1959), pp. 114 —125. url: http://www.research.ibm.com/journal/rd/032/ibmrd0302C.pdf.

[22] H. G. Rice. “Classes of recursively enumerable sets and their decision problems”. В: *Transactions of the American Mathematical Society* 74.2 (1953), с. 358—366.

[23] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009. doi: 10.1017/CBO9781139195218.

[24] Jeffrey Shallit. *A Second Course in Formal Languages and Automata Theory*. CUP, 2008.

[25] Michael Sipser. *Introduction to the Theory of Computation*. 3-е изд. Cengage Learning, 2012.

[26] L. Wittgenstein, G.H.V. Wright и G.E.M. Anscombe. *Remarks on the philosophy of psychology*. Т. 1. University of Chicago Press/B. Blackwell, 1980.

Азбучен указател