Записки по „Езици, автомати, изчислимост”

Стефан Вътев[1]

8 април 2025 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

Съдържание

1 Увод **1**

1.1 Съждително смятане 1
1.2 Предикати и квантори 4
1.3 Множества, релации, функции 7
1.4 Доказателства на твърдения 15
1.4.1 Допускане на противното 15
1.4.2 Индукция върху естествените числа 17
1.4.3 Пълна индукция върху $\mathbb{N}$ 19

2 Регулярни езици **21**

2.1 Азбуки, думи, езици 21
2.2 Регулярни изрази 25
2.3 Първи критерий за регулярност (*) 27
2.4 Уравнения от езици 28
2.5 Системи от уравнения 35
2.6 Производна на език 38
2.7 Втори критерий за регулярност 45
2.8 Хомоморфизми (*) 46
2.9 Задачи 50

3 Крайни автомати **51**

3.1 Детерминирани крайни автомати 52
3.2 Затвореност относно булеви операции 59
3.3 Каноничен автомат 62
3.4 Регулярните езици са автоматни 70
3.5 Изоморфни автомати 71
3.6 Минимален автомат 73
3.7 Минимизация 77
3.8 Недетерминирани крайни автомати 79
3.9 Фундаментална теорема 86
3.10 Автоматните езици са регулярни 89
3.11 Регулярните езици са автоматни (*) 91

3.12 Трети критерий за регулярност (*) . . . 98
3.13 Операции върху езици . . . 108
3.14 Матрично представяне на автомат (*) . . . 112
3.15 Задачи . . . 118

4 Безконтекстни езици и стекови автомати 121
4.1 Извод върху безконтекстна граматика . . . 122
4.2 Еднозначни граматики . . . 128
4.3 Апроксимации на безконтекстен език . . . 130
4.4 Фундаментална теорема за безконтекстните езици . . . 133
4.5 Лема за покачването . . . 144
4.6 Алгоритми . . . 149
4.7 Опростяване на безконтекстни граматики . . . 150
4.8 Нормална Форма на Чомски . . . 155
4.9 Проблемът за принадлежност . . . 158
4.10 Операции върху езици . . . 164
4.11 Задачи . . . 167

5 Стекови автомати 179
5.1 Недетерминирани стекови автомати . . . 179
5.2 Теорема за еквивалентност . . . 187
5.3 Сечение с регулярен език . . . 194

6 Машини на Тюринг 197
6.1 Детерминирани машини на Тюринг . . . 198
6.2 Многолентови машини на Тюринг . . . 204
6.3 Изчислими функции . . . 209
6.4 Недетерминирани машини на Тюринг . . . 212
6.5 Основни свойства . . . 216
6.6 Критерий за разрешимост . . . 223
6.7 Критерии за полуразрешимост . . . 229
6.8 Проблемът за съответствието на Пост . . . 233
6.9 Историята от всички изчисления . . . 237
6.10 Сложност . . . 243
6.11 Задачи . . . 244

Глава 1

Увод

1.1 Съждително смятане

На англ. Propositional calculus

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

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)**.

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

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.

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 $x_0 \in D$:

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

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

з) $C \setminus (A \cup B) = (C \setminus A) \cap (C \setminus B)$ и $C \setminus (A \cap B) = (C \setminus A) \cup (C \setminus B)$ — Закони на Де Морган

и) $(A \setminus B) \setminus C = (A \setminus C) \setminus (B \setminus C)$ 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 характеристично 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

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

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

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

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

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

Задача 1.4. Проверете, че:

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

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

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

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

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

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

Видове функции

Функцията $f : A \to B$ е:

// 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

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

Мощност на множество

Твърдение 1.1. Нека A и B са множества, като A е крайно, за които съществува *сюрективна* функция $f : A \to B$. Тогава $|B| \leq |A|$.

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

Вярно ли е това твърдение, ако A е безкрайно множество ?

$$g(b) \stackrel{\text{деф}}{=} a_m \text{ за } m = \min\{i < n \mid f(a_i) = b\}.$$

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. □

Твърдение 1.2. 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

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 $|X \cap Y| = 0$.

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

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 $|A_i| > 1$. □

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

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

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

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

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

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

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

□

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

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

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

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

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

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

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

от което следва, че

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

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

1.4.2 Индукция върху естествените числа

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

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

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

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

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:

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

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.

□

Глава 2

Регулярни езици

2.1 Азбуки, думи, езици

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

> The limits of my language mean the limits of my world. [24, Proposition 5.6]

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 α, β, γ, ω.

- 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 де-

финиция 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 $L = \{\omega \in \{a, b\}^\star \,:\, \omega \text{ започва с } a\}$ е пример за език над Σ.

Операции и релации върху думи

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

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

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

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

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

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

$$\omega^{\mathtt{rev}} \stackrel{\text{деф}}{=} (\beta^{\mathtt{rev}})a.$$

- Дефинираме конкатенацията на езиците A и B като

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

  Обърнете внимание, че:

$$\emptyset \cdot A = A \cdot \emptyset = \emptyset$$
$$\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A.$$

- 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 $A^0 = \{\varepsilon\}$, $A^1 = A$, $A^2 = \{abab, abba, baba, baab\}$.

- За един език A, дефинираме:

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

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

Пример 2.1. 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 $L = L^\star$.

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

Например, за $\alpha = abc$, имаме:

- $\alpha[1] = b$;
- $\alpha[1:2] = b$;
- $\alpha[1:3] = bc$;
- $\alpha[1:4] = bc$;
- $\alpha[1:1] = \varepsilon$.

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

Релации между думи

- Казваме, че думата α е **префикс** на думата β, което ще записваме като $\alpha \preceq \beta$, 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, което ще записваме като $\alpha \prec \beta$.

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

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

2.2 Регулярни изрази

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

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

Регулярните изрази могат да се опишат и по следния начин:

Това е пример за индуктивна дефиниция

- Символите $\boldsymbol{\emptyset}$, $\boldsymbol{\varepsilon}$ са регулярни изрази;
- за всяка буква $a \in \Sigma$, символът $\boldsymbol{a}$ е регулярен израз;
- ако $\boldsymbol{r_1}$ 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;

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

- Всеки регулярен израз е получен чрез крайно прилагане на горните правила.

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

Това е друг пример за индуктивна (рекурсивна) дефиниция. Нека да означим с $Reg(\Sigma)$ всики регулярни изрази над азбуката Σ. Тогава можем да си мислим за $\mathcal{L}$ като функция със сигнатура $\mathcal{L} : Reg(\Sigma) \to \mathcal{P}(\Sigma^\star)$.

- $\emptyset$ 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

$$\mathcal{L}(\boldsymbol{a}) \stackrel{\text{деф}}{=} \{a\};$$

- 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:

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

  - $L_1 \cup L_2$ е регулярен език, който се описва с регулярния израз $(\boldsymbol{r}_1 + \boldsymbol{r}_2)$. Тогава

$$\mathcal{L}(\boldsymbol{r}_1 + \boldsymbol{r}_2) \stackrel{\text{деф}}{=} \mathcal{L}(\boldsymbol{r_1}) \cup \mathcal{L}(\boldsymbol{r_2}).$$

  - $L_1 \cdot L_2$ е регулярен език, който се описва с регулярния израз $(\boldsymbol{r}_1 \cdot \boldsymbol{r}_2)$. Тогава

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

$$\mathcal{L}(\boldsymbol{r}_1 \cdot \boldsymbol{r}_2) \stackrel{\text{деф}}{=} \mathcal{L}(\boldsymbol{r}_1) \cdot \mathcal{L}(\boldsymbol{r}_2).$$

Звезда на Клини

– $L_1^\star$ е регулярен език, който се описва с регулярния израз $(\boldsymbol{r}_1)^\star$. Тогава

$$\mathcal{L}(\boldsymbol{r}_1^\star) \overset{\text{деф}}{=} \mathcal{L}(\boldsymbol{r}_1)^\star.$$

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.

В [22, 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. В този пример разглеждаме езиците от *Фигура 3.1*.

Пример 2.2. Нека да построим регулярни изрази за всеки от следните езици.

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

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

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

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

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

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

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

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

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

2.3 Първи критерий за регулярност (*)

Лема 2.1 (Слаба форма на лемата за покачването)**.** Ако L е безкраен регулярен език, то съществуват думи α, β и γ, за които:

- $\beta \neq \varepsilon$;
- $(\forall n \in \mathbb{N})[\alpha\beta^n\gamma \in L]$.

Доказателство. Индукция по построението на регулярните езици.

- Ако $L = \emptyset$, $L = \{\varepsilon\}$, $L = \{a\}$, то твърдението е изпълнено по тривиални съображения, защото тогава L не е безкраен.
- Нека $L = L_1 \cup L_2$. Нека без ограничение на общността, L_1 е безкраен език. Тогава от **(И.П.)** , съществуват думи α, $\beta \neq \varepsilon$, γ, за които $\alpha\beta^n\gamma \in L_1$ за всяко n. Тогава е ясно, че $\alpha\beta^n\gamma \in L_1 \cup L_2$ за всяко n.
- Нека $L = L_1 \cdot L_2$. Нека без ограничение на общността, L_1 е безкраен език. Тогава от **(И.П.)** , съществуват думи α, $\beta \neq \varepsilon$, γ_1, за които $\alpha\beta^n\gamma_1 \in L_1$ за всяко n. Да вземем някоя дума $\gamma_2 \in L_2$. Да положим $\gamma = \gamma_1 \cdot \gamma_2$. Тогава е ясно, че $\alpha\beta^n\gamma \in L_1 \cdot L_2$ за всяко n.
- Нека $L = L_1^\star$. Нека просто вземем *непразна* дума $\beta \in L_1$. Щом L е безкраен, то със сигурност такава дума съществува. Нека положим $\alpha = \gamma = \varepsilon$. Ясно е, че $\alpha\beta^n\gamma \in L$ за всяко n.

□

Пример 2.3. Да видим защо езикът $L = \{a^n b^n \ : \ n \in \mathbb{N}\}$ *не* е регулярен. Да допуснем, че L е регулярен. Тогава какви са вариантите за избора на думите α, $\beta \neq \varepsilon$ и γ ?

- $\beta = a^i$, за някое $i \geq 1$, или
- $\beta = b^j$, за някое $j \geq 1$, или
- $\beta = a^i b^j$, за някои $i \geq 1$ и $j \geq 1$.

И в трите случая получаваме, че $\alpha\beta^2\gamma \notin L$. В първите два случая нарушаваме равния брой срещания на a и b. В третия случай нарушаваме последователността на буквите - след срещане на b се среща отново a.

2.4 Уравнения от езици

В този раздел ще видим, че операцията звезда на Клини се оказва доста важна в нашите разглеждания. Това ще направим като разгледаме уравнения от вида $X = L \cdot X \cup M$, където X е неизвестна променлива, а L и M са произволни езици. Решение на такова уравнение е език S, за който $S = L \cdot S \cup M$. Лемата на Ардън ни показва общ метод за намиране на най-малкото решение на уравнения от този вид.

Лема 2.2 (Ардън [2]). Нека L и M са произволни езици над азбука Σ. Тогава езикът $L^\star \cdot M$ е *най-малкото решение* на уравнението

$$X = L \cdot X \cup M. \tag{2.1}$$

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

Доказателство. Първо, да видим защо $L^\star M$ е решение на уравнението (2.1). 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 (2.1). 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

Пример 2.4. Да разгледаме няколко примера.

- Уравнението

  $$X = \{a\} \cdot X \cup \{\varepsilon\}$$

  има решение $\{a\}^\star$, защото

  $$\{a\}^\star = \{a\} \cdot \{a\}^\star \cup \{\varepsilon\}.$$

  Съобразете, че това е *единственото* решение.

- Уравнението

  $$X = \{a\} \cdot X$$

  има решение $\emptyset$, защото

  $$\emptyset = \{a\} \cdot \emptyset.$$

  Съобразете, че това е *единственото* решение.

- Уравнението

  $$X = \{a, \varepsilon\} \cdot X \cup \{b\}$$

  има решение езика $\{a\}^\star \cdot \{b\}$. Друго решение на това уравнение е езика $\{a, b\}^\star$, а също и $\{a\}^\star \cdot \{b\}^+$.

В повечето учебници този резултат отсъства. Ние го смятаме за основен. Все пак може да се намери в някои учебници [20, стр. 100], [16, стр. 53], [14, стр. 60].

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

Остана да докажем, че ако $\varepsilon \notin L$, то $L^\star \cdot M$ е *единственото* решение на уравнението (2.1). Ще направим това като докажем, че всяко решение K на уравнението (2.1) е такова, че $K = L^\star \cdot M$. Достатъчно да докажем, че за всяко решение K на (2.1) 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 (2.1). Щом трябва да докажем, че $K \subseteq L^\star \cdot M$, то това означава да докажем импликацията

Да отбележим, че ако $\varepsilon \in L$, то $\Sigma^\star$ е решение на уравнението (2.1).

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

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 (2.1), 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 $K \subseteq L^\star \cdot M$. □

Следствие 2.1. За произволни регулярни изрази $\boldsymbol{r}$ и $\boldsymbol{p}$, регулярният израз $\boldsymbol{r}^\star \cdot \boldsymbol{p}$ е най-малкото решение на уравнението

$$\boldsymbol{x} = \boldsymbol{r} \cdot \boldsymbol{x} + \boldsymbol{\varepsilon}.$$

Ако $\varepsilon \notin \mathcal{L}(\boldsymbol{r})$, то това решение е и единствено.

Задача 2.1. 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*.

[16, стр. 54].

Задача 2.2. 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*.

Задача 2.3. Нека L и M са произволни езици над азбуката Σ и $\varepsilon \in L$. Докажете, че за всеки език $N \supseteq M$ над Σ, то $L^\star \cdot N$ е решение на уравнението

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

За произволни регулярни изрази $\boldsymbol{r}$ и $\boldsymbol{s}$, ще използваме записа $\boldsymbol{r} = \boldsymbol{s}$, ако $\mathcal{L}(\boldsymbol{r}) = \mathcal{L}(\boldsymbol{s})$ и $\boldsymbol{r} \leq \boldsymbol{s}$, ако $\mathcal{L}(\boldsymbol{r}) \subseteq \mathcal{L}(\boldsymbol{s})$.

Понеже *Следствие 2.1* ни показва как можем да изразим операцията звезда за регулярни изрази, сега ще разгледаме някои приложения на това наблюдение. Ще докажем някои равенства на регулярни изрази, в които участва операцията звезда.

Задача 2.4. Докажете, че изпълнено равенството:

$$\emptyset^\star = \varepsilon.$$

Упътване. Знаем, че $\emptyset^\star$ е единственото решение на уравнението

$$\boldsymbol{x} = \emptyset \cdot \boldsymbol{x} + \varepsilon.$$

Сега като заместим $\boldsymbol{x}$ с ε, получаваме равенството

$$\varepsilon = \emptyset \cdot \varepsilon + \varepsilon.$$

Така доказахме, че $\emptyset^\star = \varepsilon$. □

Задача 2.5. Докажете, че изпълнено равенството:

$$\boldsymbol{a}^\star = \boldsymbol{a}\boldsymbol{a}^\star + \varepsilon.$$

Решение. Знаем, че $\boldsymbol{a}^\star$ е единственото решение на уравнението

$$\boldsymbol{x} = \boldsymbol{a}\boldsymbol{x} + \varepsilon.$$

Като заместим променливата $\boldsymbol{x}$ с $\boldsymbol{a}^\star$ получаваме равенството:

$$\boldsymbol{a}^\star = \boldsymbol{a}\boldsymbol{a}^\star + \varepsilon.$$

□

Задача 2.6. Докажете, че изпълнено равенството:

$$(a+b)^\star = a^\star(ba^\star)^\star.$$

Основното свойство на регулярните изрази, което използваме е, че за произволен регулярен израз u е изпълнено равенството:

$$u^\star = \varepsilon + uu^\star.$$

Решение. Знаем, че $(a+b)^\star$ е единственото решение на уравнението

$$x = ax + bx + \varepsilon. \tag{2.2}$$

Оттук следва, че е достатъчно да докажем, че $a^\star(ba^\star)^\star$ е решение на уравнението (2.2), което означава, че е достатъчно да проследим равенствата:

$$\begin{aligned} a^\star(ba^\star)^\star &= \underbrace{(\varepsilon + aa^\star)}_{a^\star}(ba^\star)^\star \\ &= (ba^\star)^\star + aa^\star(ba^\star)^\star \\ &= \varepsilon + (ba^\star)(ba^\star)^\star + aa^\star(ba^\star)^\star \\ &= aa^\star(ba^\star)^\star + ba^\star(ba^\star)^\star + \varepsilon. \end{aligned}$$

□

Използвайки същата идея, можем да решим и по-общата задача.

Задача 2.7. Докажете, че за произволни регулярни изрази r и p е изпълнено равенството:

$$(r+p)^\star = r^\star(pr^\star)^\star.$$

Упътване. Тук е възможно $\varepsilon \in \mathcal{L}(r)$ или $\varepsilon \in \mathcal{L}(p)$. Това означава, че $(r+p)^\star$ е най-малкото решение на уравнението

$$x = rx + px + \varepsilon, \tag{2.3}$$

но може да не е единственото. Това не е пречка за нас. Ще докажем, че $r^\star(pr^\star)^\star$ е най-малкото решение. Ясно е, че $r^\star(pr^\star)^\star \leq (r+p)^\star$. Следователно, достатъчно е да докажем, че $r^\star(pr^\star)^\star$ е решение на (2.3). Това можем да направим както в горната задача. □

Задача 2.8. Докажете, че ако за произволни регулярни изрази r, s, t е изпълнена импликацията

$$rs = st \implies r^\star s = st^\star.$$

Упътване. Знаем, че $r^\star s$ е най-малкото решение на уравнението

$$x = rx + s. \tag{2.4}$$

Понеже имаме равенствата:

$$\begin{aligned} st^\star &= s(\varepsilon + tt^\star) \\ &= s + (st)t^\star \\ &= s + (rs)t^\star \\ &= s + r(st^\star), \end{aligned}$$

получаваме, че $st^\star$ е решение на уравнението (2.4) и оттук $r^\star s \leq st^\star$. За другата посока, $st^\star$ е най-малкото решение на уравнението

$$x = xt + s. \tag{2.5}$$

Сега разсъждаваме аналогично и така получаваме равенствата:

$$\begin{aligned} r^\star s &= (\varepsilon + r^\star r)s \\ &= s + r^\star(rs) \\ &= s + r^\star(st) \\ &= s + (r^\star s)t. \end{aligned}$$

Получихме, че $r^\star s$ е решение на уравнението (2.5) и оттук $st^\star \leq r^\star s$. □

Задача 2.9. Докажете, че ако за произволни регулярни изрази r, s е изпълнено равенството:

$$r(sr)^\star = (rs)^\star r.$$

Задача 2.10. Проверете дали са изпълнени следните равенства:

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

б) $(r^\star)^\star = r^\star$;

в) $r^\star r^\star = r^\star$;

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

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

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

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

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

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

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

2.5 Системи от уравнения

Ако разгледаме системи от няколко уравнения, то би било трудно да намерим техните решения без да имаме общ метод за това. Изглежда естествено да се запитаме дали можем да обобщим лемата на Ардън до системи от уравнения.

Теорема 2.1. Да разгледаме системата:

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

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 едно уравнение:

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

Щом $\varepsilon \notin L_{1,1}$, от лемата на Ардън веднага следва, че тази система има *единствено* решение $L_{1,1}^\star \cdot M_1$.

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:

$$X_n = L_{n,n}^\star \cdot \left(\bigcup_{i=1}^{n-1} L_{n,i} \cdot X_i \cup M_n \right), \tag{2.6}$$

което, като разкрием скобите, придобива следния вид:

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

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

Системата може да се запише по-компактно така:

$$\left|\begin{array}{l} X_1 = \bigcup\limits_{i=1}^{n} L_{1,i} \cdot X_i \cup M_1 \\ \quad \vdots \\ X_n = \bigcup\limits_{i=1}^{n} L_{n,i} \cdot X_i \cup M_n. \end{array}\right.$$

Кратко описание на този подход може да се намери и в [20, стр. 134], [14, стр. 66].

система:

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

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 езиците $S_1, \ldots, S_{n-1}$. Тогава уравнението (2.6) 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 2.1* веднага се вижда, че ако се ограничим само до регулярни езици, то единственото решение на системата ще е също съставено от регулярни езици.

Следствие 2.2. Да разгледаме системата:

$$\left|\begin{array}{lcl} X_1 & = & \bigcup\limits_{i=1}^{n} L_{1,i} \cdot X_i \cup M_1 \\ & \vdots & \\ X_n & = & \bigcup\limits_{i=1}^{n} L_{n,i} \cdot X_i \cup M_n, \end{array}\right.$$

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.

Можем да преформулираме *Следствие 2.2* като използваме регулярни изрази по следния начин.

Следствие 2.3. Да разгледаме системата:

$$\left|\begin{array}{lcl} \boldsymbol{x}_1 & = & \boldsymbol{r}_{1,1}\cdot\boldsymbol{x}_1+\boldsymbol{r}_{1,2}\cdot\boldsymbol{x}_2+\cdots+\boldsymbol{r}_{1,n}\cdot\boldsymbol{x}_n+\boldsymbol{p}_1 \\ & \vdots & \\ \boldsymbol{x}_n & = & \boldsymbol{r}_{n,1}\cdot\boldsymbol{x}_1+\boldsymbol{r}_{n,2}\cdot\boldsymbol{x}_2+\cdots+\boldsymbol{r}_{n,n}\cdot\boldsymbol{x}_n+\boldsymbol{p}_n, \end{array}\right. \tag{2.7}$$

където $\boldsymbol{r}_{i,j}$ и $\boldsymbol{p}_i$ са *регулярни изрази* , за $i,j=1,\ldots,n$, като $\varepsilon\notin\mathcal{L}(\boldsymbol{r}_{i,j})$, за всяко $i,j=1,\ldots,n$. Тогава тази система има *единствено* решение, което се описва с *регулярни изрази*.

2.6 Производна на език

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

Нека въведем следната операция за произволна буква a,

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

Бжозовски [5] описва алгоритъм за строене на автомат по регулярен израз. Вижте [20, стр. 112].

Понякога е удобно да въведем функции върху езици, които играят ролята на предикати, т.е. функции, които връщат истина / неистина. Нека още тук да въведем следното означение, което ще ни бъде полезно по-нататък:

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

Да видим как се държи операцията $a^{-1}(L)$ върху регулярните езици. За целта следваме дефиницията на регулярните езици.

Задача 2.11. Докажете, че за произволна буква a и език L е изпълнено:

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

(6) $a^{-1}(L_1 \cdot L_2) = a^{-1}(L_1) \cdot L_2 \cup \varepsilon^{[?]}(L_1) \cdot a^{-1}(L_2)$;

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

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

(9) $a^{-1}(\overline{L}) = \overline{a^{-1}(L)}$.

Лема 2.3. За всяка буква $a \in \mathbf{\Sigma}$ и регулярен език L, езикът $a^{-1}(L)$ е регулярен.

Упътване. Индукция по построението на регулярните езици като използвате *Задача 2.11*. □

Определение 2.1. За произволен език L и произволна дума ω, да положим

$$\omega^{-1}(L) \stackrel{\text{деф}}{=} \{\alpha \in \mathbf{\Sigma}^\star \ : \ \omega \cdot \alpha \in L\};$$
$$\boldsymbol{Q}_L \stackrel{\text{деф}}{=} \{\omega^{-1}(L) \ : \ \omega \in \mathbf{\Sigma}^\star\}.$$

Следващата ни задача е да покажем как можем да изразим $\omega^{-1}(L)$ с индукция по построението на регулярните езици. Доказателството на това твърдение е сложно, но по-късно ще видим, че то играе ключова роля в нашите изследвания на свойствата на регулярните езици. Да започнем с най-лесните свойства.

✍ Докажете, че за всеки език L е изпълнено:

$$L = \{\omega \in \mathbf{\Sigma}^\star \ : \ \varepsilon \in \omega^{-1}(L)\}.$$

✍ Освен това, докажете, че за произволни думи α и β е изпълнено:

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

✍ Докажете с индукция по дължината на думата ω, че $\omega^{-1}(L)$ е регулярен за всеки регулярен език L.

Твърдение 2.1. За произволна дума ω и произволни езици L и M са изпълнени равенствата:

$$\omega^{-1}(L \cup M) = \omega^{-1}(L) \cup \omega^{-1}(M); \tag{2.8}$$

$$\omega^{-1}(L \cap M) = \omega^{-1}(L) \cap \omega^{-1}(M); \tag{2.9}$$

$$\omega^{-1}(L \setminus M) = \omega^{-1}(L) \setminus \omega^{-1}(M). \tag{2.10}$$

Упътване. Ще разгледаме само (2.8), защото другите равенства се доказват по подобен начин. Достатъчно е да проследим еквивалентностите:

$$\begin{aligned} \alpha \in \omega^{-1}(L \cup M) &\Leftrightarrow \omega\alpha \in L \cup M \\ &\Leftrightarrow \omega\alpha \in L \ \vee \ \omega\alpha \in M \\ &\Leftrightarrow \alpha \in \omega^{-1}(L) \cup \omega^{-1}(M). \end{aligned}$$

□

Производната на конкатенацията на два езика дава основната идея как да изразим и другите производни на езици.

Твърдение 2.2. За произволна дума ω и произволни езици L и M са изпълнени равенствата:

$$\omega^{-1}(L \cdot M) = \omega^{-1}(L) \cdot M \cup \bigcup_{\omega=\omega_1\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M); \tag{2.11}$$

Упътване. Да започнем с включването ($\subseteq$). Да видим какво представляват думите α принадлежащи на езика $\omega^{-1}(L \cdot M)$. Да разгледаме произволни думи $\lambda \in L$ и $\mu \in M$, за които $\omega\alpha = \lambda\mu$. Тогава за думата α имаме следните случаи:

- Думата ω „изяжда" само част от λ, т.е. за някой индекс i имаме:

$$\underbrace{\lambda[:i]}_{\omega} \cdot \underbrace{\lambda[i:]}_{\alpha} \cdot \mu .$$

  Алгебрично този случай се представя като $\alpha \in \omega^{-1}(L) \cdot M$.

- Думата ω „изяжда" цялата дума λ и част от μ, т.е. за някой индекс j имаме:

$$\underbrace{\lambda \cdot \mu[:j]}_{\omega} \cdot \underbrace{\mu[j:]}_{\alpha} .$$

  С други думи, има разделяне на ω на две части като $\omega = \omega_1 \cdot \omega_2$ и

$$\underbrace{\lambda}_{\omega_1} \cdot \underbrace{\mu[:j]}_{\omega_2} \cdot \underbrace{\mu[j:]}_{\alpha} .$$

  Алгебрично този случай се представя така:

$$\alpha \in \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M).$$

Така доказахме включването:

$$\omega^{-1}(L \cdot M) \subseteq \omega^{-1}(L) \cdot M \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M).$$

За обратната посока, нека

$$\alpha \in \omega^{-1}(L) \cdot M \cup \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M).$$

Тук имаме два случая. Ако $\alpha \in \omega^{-1}(L) \cdot M$, то α може да се представи като $\alpha = \alpha_1\alpha_2$ и $\omega\alpha_1 \in L$ и $\alpha_2 \in M$. Ясно, че $\omega \cdot (\alpha_1 \cdot \alpha_2) \in L \cdot M$ и оттук $\alpha \in \omega^{-1}(L \cdot M)$.

Второ, ако $\alpha \in \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M)$, то за някое разбиване на ω като $\omega = \omega_1\omega_2$ имаме, че $\omega_1 \in L$ и $\omega_2\alpha \in M$. Тогава е ясно, че $(\omega_1\omega_2)\alpha \in L \cdot M$ и оттук $\alpha \in \omega^{-1}(L \cdot M)$. Така заключаваме, че

$$\omega^{-1}(L) \cdot M \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M) \subseteq \omega^{-1}(L \cdot M).$$

□

Следващата стъпка е да видим как можем да преминем от конкатенация на два езика до конкатенация на произволен краен брой езици. Да обърнем

внимание, че за $L = M$ можем да получим представянето:

$$\omega^{-1}(L^2) = \omega^{-1}(L) \cdot L \cup \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(L)$$

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

Лесно можем да съобразим, че горното равенство се обобщава за произволно n. Този случай ще ни трябва за да изразим и звездата на Клини на език.

Твърдение 2.3. За произволна дума ω, произволен език L и естествено число n са изпълнени равенствата:

$$\omega^{-1}(L^{n+1}) = \bigcup_{n=\ell+r} \bigcup_{\omega=\omega_1\omega_2} \omega_1^{[?]}(L^\ell) \cdot \omega_2^{-1}(L) \cdot L^r. \tag{2.12}$$

Упътване. Да започнем с включването ($\subseteq$). Да видим какво представляват думите α принадлежащи на езика $\omega^{-1}(L^{n+1})$. Да разгледаме произволни думи $\lambda_1, \ldots, \lambda_{n+1}$ принадлежащи на езика L, за които $\omega\alpha = \lambda_1 \cdots \lambda_{n+1}$. Нека ω „изяжда" напълно $\lambda_1, \ldots, \lambda_\ell$, за някое $\ell \leq n$, и част от $\lambda_{\ell+1}$. Това означава, че съществува индекс i, за който имаме разбиването:

$$\underbrace{\lambda_1 \cdots \lambda_\ell \cdot \lambda_{\ell+1}[:i]}_{\omega} \cdot \underbrace{\lambda_{\ell+1}[i:] \cdot \lambda_{\ell+2} \cdots \lambda_{n+1}}_{\alpha}.$$

Оттук имаме разбиване на ω на две части като $\omega = \omega_1\omega_2$, за които:

$$\underbrace{\lambda_1 \cdots \lambda_\ell}_{\omega_1} \cdot \underbrace{\lambda_{\ell+1}[:i]}_{\omega_2} \cdot \underbrace{\lambda_{\ell+1}[i:] \cdot \lambda_{\ell+2} \cdots \lambda_{n+1}}_{\alpha}.$$

Понеже индексът ℓ също е произволен, можем да обобщим всичко така:

$$\alpha \in \bigcup_{n=\ell+r} \bigcup_{\omega=\omega_1\omega_2} \omega_1^{[?]}(L^\ell) \cdot \omega_2^{-1}(L) \cdot L^r.$$

Така доказахме включването ($\subseteq$). За обратното включване, нека сега

$$\alpha \in \bigcup_{n=\ell+r} \bigcup_{\omega=\omega_1\omega_2} \omega_1^{[?]}(L^\ell) \cdot \omega_2^{-1}(L) \cdot L^r.$$

Това означава, че съществува индекс $\ell \leq n$ и разбиване на ω на две части като $\omega = \omega_1\omega_2$, за които $\omega_1 \in L^\ell$ и $\alpha \in \omega_2^{-1}(L) \cdot L^{n-\ell}$. Оттук получаваме, че $\omega_2\alpha \in L^{n+1-\ell}$ и следователно $\omega_1\cdot\omega_2\cdot\alpha \in L^\ell \cdot L^{n+1-\ell}$. Заключаваме, че $\alpha \in \omega^{-1}(L^{n+1})$. $\square$

Твърдение 2.4. За произволна *непразна* дума ω и произволен език L е изпълнено равенството:

$$\omega^{-1}(L^\star) = \bigcup_{\omega=\omega_1\omega_2} \omega_1^{[?]}(L^\star) \cdot \omega_2^{-1}(L) \cdot L^\star. \tag{2.13}$$

Понеже $L^\star = \bigcup_{n\geq 0} L^n$, то този случай не крие изненади. ✍ Единствено трябва да съобразим, че понеже ω е *непразна* дума, то $\omega^{-1}(L^\star) = \omega^{-1}(L^+)$.

Упътване. Достатъчно е да проследим веригата от равенства:

$$\begin{aligned}
\omega^{-1}(L^\star) &= \omega^{-1}(\bigcup_n L^{n+1}) \\
&= \bigcup_n \omega^{-1}(L^{n+1}) \\
&= \bigcup_n \left[\bigcup_{n=\ell+r} \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L^\ell) \cdot \omega_2^{-1}(L) \cdot L^r \right] \\
&= \bigcup_\ell \bigcup_r \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L^\ell) \cdot \omega_2^{-1}(L) \cdot L^r \\
&= \bigcup_{\omega=\omega_1\cdot\omega_2} \bigcup_\ell \bigcup_r \omega_1^{[?]}(L^\ell) \cdot \omega_2^{-1}(L) \cdot L^r \\
&= \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(\bigcup_\ell L^\ell) \cdot \omega_2^{-1}(L) \cdot (\bigcup_r L^r) \\
&= \bigcup_{\omega=\omega_1\cdot\omega_2} \omega_1^{[?]}(L^\star) \cdot \omega_2^{-1}(L) \cdot L^\star
\end{aligned}$$

□

Твърдение 2.5. За всяка дума $\omega \in \Sigma^\star$ и регулярен език L, езикът $\omega^{-1}(L)$ е регулярен.

Упътване. Индукция по построението на регулярните езици като използвате предишните няколко твърдения. □

Вижте[20, стр. 142].

Оказва се, че е интересно да се изследва мощността на множеството Q_L.

Теорема 2.2. Ако L е регулярен, то Q_L е крайно множество.

Доказателство. Индукция по построението на регулярните езици.

- Нека $L = \emptyset$. Ясно е, че $\boldsymbol{Q}_L = \{\emptyset\}$.
- Нека $L = \{\varepsilon\}$. Ясно е, че $\boldsymbol{Q}_L = \{\{\varepsilon\}, \emptyset\}$.
- Нека $L = \{a\}$. Ясно е, че $\boldsymbol{Q}_L = \{\{a\}, \{\varepsilon\}, \emptyset\}$.
- Нека сега L и M са регулярни езици. От **(И.П.)** имаме, че $\boldsymbol{Q}_L$ и $\boldsymbol{Q}_M$ са крайни. Да положим $\boldsymbol{Q}_L = \{K_0, \ldots, K_\ell\}$ и $\boldsymbol{Q}_M = \{N_0, \ldots, N_m\}$.

Да видим защо $\boldsymbol{Q}_{L \cup M}$ е крайно. От (2.8) имаме представянето:

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

Това означава, че за всяка дума ω, съществуват индекси $i \leq n$ и $j \leq m$, за които за които $\omega^{-1}(L \cup M) = K_i \cup N_j$. С други думи, всеки елемент на $\boldsymbol{Q}_{L \cup M}$ се характеризира с двойка $\langle i, j \rangle \in \{0, \ldots, \ell\} \times \{0, \ldots, m\}$. Това означава, че

$$|\boldsymbol{Q}_{L \cup M}| \leq |\boldsymbol{Q}_L \times \boldsymbol{Q}_M| = |\boldsymbol{Q}_L| \cdot |\boldsymbol{Q}_M|.$$

Щом $\boldsymbol{Q}_L$ и $\boldsymbol{Q}_M$ са крайни, то $\boldsymbol{Q}_{L \cup M}$ също е крайно.

Да видим защо $\boldsymbol{Q}_{L \cdot M}$ е крайно. От (2.11) имаме представянето:

$$\omega^{-1}(L \cdot M) = \omega^{-1}(L) \cdot M \cup \bigcup_{\omega = \omega_1 \cdot \omega_2} \omega_1^{[?]}(L) \cdot \omega_2^{-1}(M).$$

Това означава, че за всяка дума ω, съществува индекс $i \leq \ell$ и подмножество $J \subseteq \{0, \ldots, m\}$, за които

$$\omega^{-1}(L \cdot M) = K_i \cdot M \cup \bigcup_{j \in J} N_j.$$

С други думи, всеки елемент на $Q_{L_1 \cdot L_2}$ се характеризира с двойка от вида $\langle i, J \rangle \in \{0, \ldots, \ell\} \times \mathcal{P}(\{0, \ldots, m\})$. Това означава, че

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

Щом $\boldsymbol{Q}_L$ и $\boldsymbol{Q}_M$ са крайни, то $\boldsymbol{Q}_{L \cdot M}$ също е крайно.

Остана да видим защо $\boldsymbol{Q}_{L^\star}$ е крайно. От (2.13) имаме представянето:

$$\omega^{-1}(L^\star) = \bigcup_{\omega_1 \cdot \omega_2 = \omega} \omega_1^{[?]}(L^\star) \cdot \omega_2^{-1}(L) \cdot L^\star.$$

Това означава, че за всеки елемент на $\boldsymbol{Q}_{L^\star}$ съществува крайно множество $J \subseteq \{0, \ldots, \ell\}$, за което

$$\omega^{-1}(L^\star) = \bigcup_{j \in J} K_j \cdot L^\star.$$

С други думи, всеки елемент на $\boldsymbol{Q}_{L^\star}$ се характеризира с елемент $J \in \mathcal{P}(\{0, \ldots, \ell\})$.

Заключаваме, че

$$|\boldsymbol{Q}_{L^\star}| \leq |\mathcal{P}(\boldsymbol{Q}_L)| = 2^{|\boldsymbol{Q}_L|}.$$

Щом $\boldsymbol{Q}_L$ е крайно, то $\boldsymbol{Q}_{L^\star}$ също е крайно.

□

По-късно ще видим, че това свойство точно харектеризира регулярните езици, т.е. ще докажем, че един език е регулярен точно тогава, когато $\boldsymbol{Q}_L$ е крайно.

2.7 Втори критерий за регулярност

За да докажем, че един език L не е регулярен можем да приложим *Теорема 2.2* като докажем, че $\boldsymbol{Q}_L$ е безкрайно множество. Обърнете внимание, че не е нужно да опишем всички елементи на $\boldsymbol{Q}_L$. Достатъчно е само да докажем, че има безкрайно много такива елементи.

Пример 2.5. За езика $L = \{a^n b^n \ : \ n \in \mathbb{N}\}$ имаме, че за произволни естествени числа k и m е изпълнено следното свойство:

Сравнете с *Пример 2.3*.

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

Проверете, че $(a^k)^{-1}(L) = \{a^n b^{n+k} \ : \ n \in \mathbb{N}\}$, за всяко $k \in \mathbb{N}$. Така получаваме, че $\boldsymbol{Q}_L$ е безкрайно. Заключаваме, че този език **не** е регулярен.

Пример 2.6. За езика $L = \{a^{n^2} \ : \ n \in \mathbb{N}\}$ 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 2.7. За езика $L = \{a^{n!} \ : \ n \in \mathbb{N}\}$ 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.

2.8 Хомоморфизми (*)

Определение 2.2. 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 $h : \Sigma_1^\star \to \Sigma_2^\star$, казваме, че **образът** на $L \subseteq \Sigma_1^\star$ относно h е множеството $h(L) \stackrel{\text{деф}}{=} \{h(\omega) \in \Sigma_2^\star \,:\, \omega \in L\}$.

Твърдение 2.6. 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:

$$h(L_1 \cup L_2) = h(L_1) \cup h(L_2).$$

Тук няма нужда h да бъде хомоморфизъм.

Доказателство. Достатъчно е да проследим веригата от равенства:

$$\begin{aligned} h(L_1 \cup L_2) &= \{h(\omega) \,:\, \omega \in L_1 \vee \omega \in L_2\} \\ &= \{h(\omega) \,:\, \omega \in L_1\} \cup \{h(\omega) \,:\, \omega \in L_2\} \\ &= h(L_1) \cup h(L_2). \end{aligned}$$

□

Твърдение 2.7. 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:

$$\begin{aligned} h(\bigcup_n L_n) &= \{h(\alpha) \,:\, (\exists n)[\alpha \in L_n]\} \\ &= \bigcup_n \{h(\alpha) \,:\, \alpha \in L_n\} \\ &= \bigcup_n h(L_n). \end{aligned}$$

□

Твърдение 2.8. 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:

$$\begin{aligned} h(L_1 \cdot L_2) &= \{h(\alpha_1\alpha_2) \,:\, \alpha_1 \in L_1 \,\&\, \alpha_2 \in L_2\} \\ &= \{h(\alpha_1) \cdot h(\alpha_2) \,:\, \alpha_1 \in L_1 \,\&\, \alpha_2 \in L_2\} \qquad \text{// } h \text{ е хомоморфизъм} \\ &= \{\omega_1\omega_2 \,:\, \omega_1 \in h(L_1) \,\&\, \omega_2 \in h(L_2)\} \\ &= h(L_1) \cdot h(L_2). \end{aligned}$$

□

Твърдение 2.9. 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:

$$\begin{aligned} h(L^{n+1}) &= h(L^n \cdot L) \\ &= h(L^n) \cdot h(L) && \text{// от } \textit{Твърдение 2.8} \\ &= h(L)^n \cdot h(L) && \text{// от } \textbf{(И.П.)} \\ &= h(L)^{n+1}. \end{aligned}$$

□

Лема 2.4. 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 2.4 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 2.6* 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 2.8* 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 предпо-

ложение, че $h(L)$ е регулярен език. Имаме равенствата:

$$\begin{aligned} h(L^\star) &= h(\bigcup_n L^n) && \text{// деф. на звезда на Клини} \\ &= \bigcup_n h(L^n) && \text{// от } \textit{Твърдение 2.7} \\ &= \bigcup_n h(L)^n && \text{// от } \textit{Твърдение 2.9} \\ &= h(L)^\star. && \text{// деф. на звезда на Клини} \end{aligned}$$

Щом от **(И.П.)** имаме, че $h(L)$ е регулярен език, то $h(L^\star) = h(L)^\star$ също е регулярен език.

□

Пример 2.8. Да разгледаме езика $L = \{c^k a^n c^\ell b^n c^m \ : \ k, n, \ell, m \in \mathbb{N}\}$, в който е „закодирана" нерегулярна част. Да видим как 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 езикът $h(L) = \{a^n b^n \ : \ n \in \mathbb{N}\}$ е регулярен. Достигнахме до противоречие.

Задача 2.12. Докажете, че езикът

$$L \stackrel{\text{деф}}{=} \{\omega \in \{a, b\}^\star \ : \ \text{след всяко срещане на } a \text{ веднага следва } b\}$$

е регулярен.

Упътване. Да разгледаме $h(a) \stackrel{\text{деф}}{=} ab$ и $h(b) \stackrel{\text{деф}}{=} b$. Тогава $h(\{a, b\}^\star) = L$. □

Забележка. И така, вече знаем, че ако L е регулярен и h е хомоморфизъм, то $h(L)$ е регулярен. В общия случай, не можем да твърдим, че имаме обратната посока, т.е. съществува хомоморфизъм h, нерегулярен език L, за които $h(L)$ 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 $L = \{a^n b^n \ : \ n \in \mathbb{N}\}$ имаме, че $h(L) = \{a^{2n} \ : \ n \in \mathbb{N}\}$, който е регулярен.

2.9 Задачи

Задача 2.13. Докажете, че ако L е регулярен език, то $\texttt{Pref}(L)$ е регулярен.

Задача 2.14. Докажете, че ако L е регулярен език, то $\texttt{Suff}(L)$ е регулярен.

Задача 2.15. Докажете, че ако L е регулярен език, то $\texttt{Rev}(L)$ е регулярен.

Например, $101 \in L$, защото съдържа по веднъж 10 и 01. $1010 \notin L$, защото съдържа два пъти 10 и само веднъж 01.

Задача 2.16. Докажете, че L е регулярен език, където

$$L = \{\omega \in \{0,1\}^\star \,:\, \omega \text{ съдържа равен брой поднизове } 01 \text{ и } 10\}.$$

Задача 2.17. Докажете, че езикът $L = \{a^n b^n \,:\, n \in \mathbb{N}\}$ е *единственото* решение на уравнението

$$X = \{a\} \cdot X \cdot \{b\} \cup \{\varepsilon\}.$$

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

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

Задача 2.18. Докажете, че:

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

✍ Докажете с индукция по построението на регулярните езици като използвате *Задача 2.18*.

Задача 2.19. За всеки език M и регулярен език L, езикът $M^{-1}(L)$ е регулярен.

Глава 3

Крайни автомати

Сега ще разгледаме модел на абстрактна машина, с чиято помощ отново ще опишем регулярните езици. Този нов подход ще ни помогне да решим някои трудни задачи за регулярни езици.

3.1 Детерминирани крайни автомати

На англ. *automaton* в единствено число и *automata* в множ. число.

Ще започнем с най-простия машинен модел в този курс - този на детерминираните крайни автомати.

Определение 3.1. Детерминиран краен автомат (ДКА) е петорка

$$\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle, \text{ където:}$$

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

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

- $q_0 = s$, началното състояние на автомата;
- $\delta(q_i, a_i) = q_{i+1}$, за всеки индекс $i < n$;
- $q_n \in F$.

Казваме, че $\mathcal{A}$ **разпознава** езика L, ако $\mathcal{A}$ разпознава точно думите от L, т.е. $L = \{\omega \in \Sigma^\star \ : \ \mathcal{A} \text{ разпознава } \omega\}$. 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 := s
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 $\boldsymbol{\delta}$, често е удобно да разглеждаме функция $\boldsymbol{\delta}^\star : \boldsymbol{Q} \times \boldsymbol{\Sigma}^\star \to \boldsymbol{Q}$, която е дефинирана за всяко $q \in \boldsymbol{Q}$ и $\omega \in \boldsymbol{\Sigma}^\star$ по следния начин:

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

Лесно се съобразява, че една дума ω се *разпознава* от автомата $\mathcal{A}$ точно тогава, когато $\boldsymbol{\delta}^\star(\boldsymbol{s}, \omega) \in \boldsymbol{F}$.

Определение 3.2. За произволен ДКА $\mathcal{A}$, дефинираме

$$\mathcal{L}(\mathcal{A}) \stackrel{\text{деф}}{=} \{\, \omega \in \boldsymbol{\Sigma}^\star \;:\; \boldsymbol{\delta}^\star(\boldsymbol{s}, \omega) \in \boldsymbol{F} \,\}.$$

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 $q \in \boldsymbol{Q}$,

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

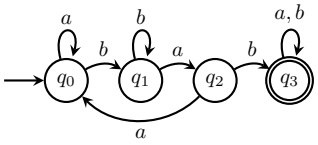
В частност, $\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(\boldsymbol{s})$. В този ред на мисли, при някои по-сложни конструкции, ще бъде полезно да използваме произволно множество от състояния като финални. Поради тази причина е удобно да въведем означението

$$\mathcal{L}_{\mathcal{A}}(q, P) \stackrel{\text{деф}}{=} \{\omega \in \boldsymbol{\Sigma}^\star \;:\; \boldsymbol{\delta}^\star(q, \omega) \in P\},$$

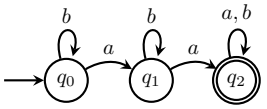
където $q \in \boldsymbol{Q}$ и $P \subseteq \boldsymbol{Q}$. В частност, $\mathcal{L}(\mathcal{A}) = \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, \boldsymbol{F})$. В случая, когато $P = \{p\}$, ще пишем $\mathcal{L}_{\mathcal{A}}(q, p)$ вместо $\mathcal{L}_{\mathcal{A}}(q, \{p\})$. Лесно се съобразява, че:

$$\mathcal{L}(\mathcal{A}) = \bigcup_{p \in \boldsymbol{Q}} \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \cdot \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}).$$

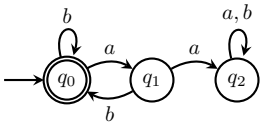
Фигура 3.1: Примери за детерминирани крайни автомати.



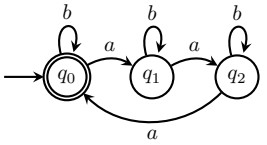
(а) Автомат разпознаващ думите, които съдържат bab.



(б) Автомат разпознаващ думите, които съдържат поне две срещания на буквата a.



(в) Автомат разпознаващ думите, в които веднага след всяко срещане на буквата a следва поне едно срещане на буквата b.



(г) Автомат разпознаващ думите, в които броят на срещанията на буквата a се дели на три.

Твърдение 3.1. Нека $\mathcal{A}$ е ДКА. Тогава за всяко състояние q и произволни думи α и β е изпълнено равенството:

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

Упътване. Доказателството протича с индукция по дължината на β.

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

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

- 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:

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

□

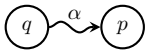
Забележка. Формално погледнато, доказателството на *Твърдение 3.1* протича по следната схема:

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

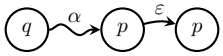
където свойството P е дефинирано така:

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

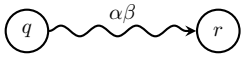
(а) Да започнем с изчислението:



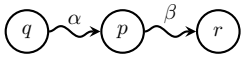
(б) Базата на индукцията ни казва, че то може да се представи и така:



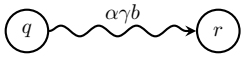
(в) Сега да разгледаме следното изчисление, където думата β има дължина n:



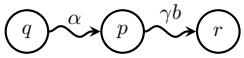
(г) Индукционното предположение ни казва, че това изчисление може да се разбие така:



(д) Накрая, нека разгледаме следното изчисление, където думата γ има дължина n:



(е) Индукционната стъпка ни казва, че това изчисление може да се разбие така:



Примерни задачи

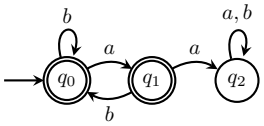
В този раздел ще разгледаме няколко примера за автоматни езици и ще видим как можем да докажем, че конкретен автомат разпознава даден език.

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.

За момента нямаме общ метод, с който да докажем, че даденият автомат разпознава точно съответния език.

Задача 3.1. Докажете, че езикът L е автоматен, където:

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



Фигура 3.3: Автомат разпознаващ думите, които не съдържат 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 3.3*. По-късно ще разгледаме общ метод за строене на автомат по език.

Доказателство. Да разгледаме $\mathcal{A} = \langle \boldsymbol{\Sigma}, \boldsymbol{Q}, \boldsymbol{s}, \boldsymbol{\delta}, \boldsymbol{F} \rangle$ с функция на преходите описана на *Фигура 3.3*. 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:

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

(2) ако $\boldsymbol{\delta}^\star(q_0, \alpha) = q_1$, 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 (1) и (2) са изпълнени 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 $\boldsymbol{\delta}^\star(q_0, \beta x) = q_0 = \boldsymbol{\delta}(\boldsymbol{\delta}^\star(q_0, \beta), x)$. Според дефиницията на функцията $\boldsymbol{\delta}$, $x = b$ и $\boldsymbol{\delta}^\star(q_0, \beta) \in \{q_0, q_1\}$. 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 $\boldsymbol{\delta}^\star(q_0, \beta x) = q_1 = \boldsymbol{\delta}(\boldsymbol{\delta}^\star(q_0, \beta), x)$. Според дефиницията на $\boldsymbol{\delta}$, $x = a$ и $\boldsymbol{\delta}^\star(q_0, \beta) = q_0$. 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 $\boldsymbol{\delta}^\star(q_0, \alpha) \in$

$\boldsymbol{F} = \{q_0, q_1\}$ 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 $\mathcal{L}(\mathcal{A}) \subseteq L$. Сега ще докажем другата посока, т.е. включването $L \subseteq \mathcal{L}(\mathcal{A})$. Това означава да докажем импликацията $(\forall \alpha \in \Sigma^\star)[\alpha \in L \;\Rightarrow\; \boldsymbol{\delta}^\star(q_0, \alpha) \in \boldsymbol{F}]$, която разгледана в контрапозиция, е еквивалентна на

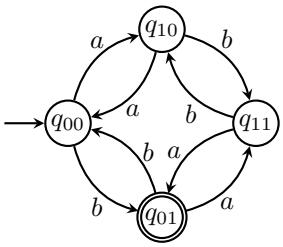
Да напомним, че от съждителното смятане знаем, че $p \to q \equiv \neg q \to \neg p$.

$$(\forall \alpha \in \Sigma^\star)[\boldsymbol{\delta}^\star(q_0, \alpha) \notin \boldsymbol{F} \;\Rightarrow\; \alpha \notin L]. \tag{3.1}$$

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

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

Използвайки свойство (2) от по-горе, понеже $\boldsymbol{\delta}^\star(q_0, \beta) = q_1$, 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 3.1, а следователно и посоката $L \subseteq \mathcal{L}(\mathcal{A})$. □



Фигура 3.4: $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} L$

Задача 3.2. Докажете, че езикът L е автоматен, където:

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

Да напомним, че $|\omega|_a \stackrel{\text{деф}}{=}$ броят на срещанията на буквата a в думата ω. Тази задача е важна, защото в явен вид показва как кодираме информация в състоянията на автомата, а именно информацията за остатъците при деление на 2 на $|\omega|_a$ и $|\omega|_b$.

Упътване. Разгледайте детерминирания автомат $\mathcal{A}$ на *Фигура 3.4*. Докажете с индукция по дължината на думата ω, че:

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

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

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

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

Оттук направете извода, че за произволна дума ω,

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

□

Представяне на числа

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,

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

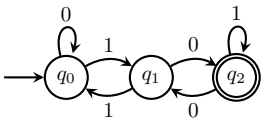
Задача 3.3. Докажете, че езикът L е автоматен, където:

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

Упътване. 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:

$$(\forall \alpha \in \{0,1\}^\star)(\forall i < 3)[\, \overline{\alpha}_{(2)} \equiv i \bmod 3 \;\Leftrightarrow\; \boldsymbol{\delta}^\star(q_0, \alpha) = q_i \,], \tag{3.2}$$

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



Фигура 3.5: Автоматът $\mathcal{A}$.

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

Да разгледаме твърденията:

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

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

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

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?) Нека $|\alpha| > 0$, т.е. $\alpha = \beta x$.

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

- $x = 0$. Тогава, по дефиницията на $\boldsymbol{\delta}$, $\boldsymbol{\delta}(q_1, 0) = q_2$ и следователно, $\boldsymbol{\delta}^\star(q_0, \beta) = q_1$. По **(И.П.)** за (2) с β получаваме, че $\overline{\beta}_{(2)} \equiv 1 \bmod 3$. Тогава, $\overline{\beta 0}_{(2)} \equiv 2 \mod 3$. Така доказахме, че

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

- $x = 1$. Тогава, по дефиницията на $\boldsymbol{\delta}$, $\boldsymbol{\delta}(q_2, 1) = q_2$ и следователно, $\boldsymbol{\delta}^\star(q_0, \beta) = q_2$. По **(И.П.)** за (3) с β имаме, че $\overline{\beta}_{(2)} \equiv 2 \bmod 3$. Тогава, $\overline{\beta 1}_{(2)} \equiv 2 \mod 3$. Така доказахме, че

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

✍ Довършете доказателствата на свойства (1) и (2)!

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

- $x = 0$. 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 $\boldsymbol{\delta}^\star(q_0, \beta x) = q_1$.

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

✍ Защо?

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

$$(\forall \alpha \in \{0,1\}^\star)(\forall i < 3)[\, \overline{\alpha}_{(2)} \equiv i \pmod 3 \;\Leftrightarrow\; \boldsymbol{\delta}^\star(q_0, \alpha) = q_i \,],$$

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

3.2 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 3.2. Класът на автоматните езици е затворен относно операцията *сечение*. Това означава, че ако L_1 и L_2 са два произволни автоматни езика над азбуката Σ, то $L_1 \cap L_2$ също е автоматен език.

Доказателство. Да разгледаме два автомата

$$\mathcal{A}_1 = \langle \Sigma, Q_1, s_1, \delta_1, F_1 \rangle \text{ и } \mathcal{A}_2 = \langle \Sigma, Q_2, s_2, \delta_2, F_2 \rangle.$$

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

- $Q \stackrel{\text{деф}}{=} Q_1 \times Q_2$;
- $\delta : Q_1 \times Q_2 \times \Sigma \to Q_1 \times Q_2$, като за всяко $\langle r_1, r_2 \rangle \in Q$ и всяко $a \in \{0,1\}$,

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

- $s \stackrel{\text{деф}}{=} \langle s_1, s_2 \rangle$;
- $F \stackrel{\text{деф}}{=} \{\langle r_1, r_2 \rangle \,:\, r_1 \in F_1 \,\&\, r_2 \in F_2\} = F_1 \times F_2$

Трябва да докажем, че за всяка дума $\omega \in \{0,1\}^\star$ е изпълнено, че:

$$(\forall p \in Q_1)(\forall q \in Q_2)[\, \delta^\star(\langle p, q \rangle, \omega) = \langle \delta_1^\star(p, \omega), \delta_2^\star(q, \omega) \rangle \,]. \tag{3.3}$$

Ще докажем *Свойство 3.3* с индукция по дължината на думата ω.

- Нека $|\omega| = 0$, т.е. $\omega = \varepsilon$. 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 3.3* е изпълнено за думи ω с дължина n.
- Нека $|\omega| = n + 1$, т.е. $\omega = \beta a$ и $|\beta| = n$. Тогава:

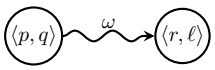
$$\begin{aligned} \delta^\star(\langle p, q \rangle, \beta a) &= \delta(\delta^\star(\langle p, q \rangle, \beta), a) && \text{// деф. на } \delta^\star \\ &= \delta(\langle \delta_1^\star(p, \beta), \delta_2^\star(q, \beta) \rangle, a) && \text{// от } \textbf{(И.П.)} \\ &= \langle \delta_1(\delta_1^\star(p,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

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

(а) Нека имаме изчисление на думата ω в автомата $\mathcal{A}_1$:

(б) и отделно изчисление на думата ω в автомата $\mathcal{A}_2$:

(в) Можем да комбинираме двете изчисления на думата ω в едно изчисление и така в автомата $\mathcal{A}$ получаваме:



Използвайки *Свойство 3.3* лесно можем да докажем, че

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

Имаме следните еквивалентности:

$$\begin{aligned}
\omega \in \mathcal{L}(\mathcal{A}) &\Leftrightarrow \boldsymbol{\delta}^\star(\langle \boldsymbol{s}_1, \boldsymbol{s}_2\rangle, \omega) \in \boldsymbol{F}_1 \times \boldsymbol{F}_2 && \text{// деф. на } \mathcal{A} \\
&\Leftrightarrow \langle \boldsymbol{\delta}_1^\star(\boldsymbol{s}_1, \omega), \boldsymbol{\delta}_2^\star(\boldsymbol{s}_2, \omega)\rangle \in \boldsymbol{F}_1 \times \boldsymbol{F}_2 && \text{// от (3.3)} \\
&\Leftrightarrow \boldsymbol{\delta}_1^\star(\boldsymbol{s}_1, \omega) \in \boldsymbol{F}_1 \;\&\; \boldsymbol{\delta}_2^\star(\boldsymbol{s}_2, \omega) \in \boldsymbol{F}_2 \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{A}_1) \;\&\; \omega \in \mathcal{L}(\mathcal{A}_2) \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2).
\end{aligned}$$

□

Твърдение 3.3. Класът на автоматните езици са затворени относно операцията *допълнение*, т.е. ако L е автоматен език, то $\boldsymbol{\Sigma}^\star \setminus L$ също е автоматен език.

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

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

$$\mathcal{A}' = \langle \boldsymbol{\Sigma}, \boldsymbol{Q}, \boldsymbol{s}, \boldsymbol{\delta}, \boldsymbol{Q} \setminus \boldsymbol{F}\rangle,$$

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

Твърдение 3.4. Класът на автоматните езици е затворен относно операцията *обединение*. Това означава, че ако L_1 и L_2 са два произволни автоматни езика, то $L_1 \cup L_2$ също е автоматен език.

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

Упътване. Първият подход е да използваме конструкцията на автомата $\mathcal{A}$ от *Твърдение 3.2*, с единствената разлика, че тук избираме финалните състояния да бъдат елементите на множеството

$$\begin{aligned}
\boldsymbol{F} &\stackrel{\text{деф}}{=} \{\langle q_1, q_2\rangle \in \boldsymbol{Q}_1 \times \boldsymbol{Q}_2 \;:\; q_1 \in \boldsymbol{F}_1 \;\vee\; q_2 \in \boldsymbol{F}_2\} \\
&= \boldsymbol{F}_1 \times \boldsymbol{Q}_2 \cup \boldsymbol{Q}_1 \times \boldsymbol{F}_2.
\end{aligned}$$

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

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

□

Като приложение на наученото тук, нека отново да разгледаме *Задача 3.2* като този път ще разбием задачата на по-прости задачи и оттам с конструкцията за сечение ще получим решение на *Задача 3.2*.

Задача 3.4. Докажете, че езикът L е автоматен, където:

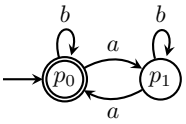
$$L = \{\omega \in \{a,b\}^\star \,:\, |\omega|_a \equiv 0 \bmod 2 \;\&\; |\omega|_b \equiv 1 \bmod 2\}.$$

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

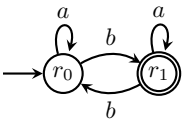
$$L_1 = \{\omega \in \{a,b\}^\star \,:\, |\omega|_a \equiv 0 \bmod 2\}$$
$$L_2 = \{\omega \in \{a,b\}^\star \,:\, |\omega|_b \equiv 1 \bmod 2\}.$$

Да разгледаме автоматите $\mathcal{A}_1$ и $\mathcal{A}_2$ от *Фигура 3.7а* и *Фигура 3.7б*. Лесно се съобразява, че те разпознават съответно езиците L_1 и L_2. Ясно е, че $L = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$. Ще построим автомат $\mathcal{A}$ като използваме конструкцията от *Твърдение 3.2*. Така финалното състояние на $\mathcal{A}$ ще бъде наредената двойка $\langle p_0, r_1\rangle$, 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 $\mathcal{A}_2$. Функцията на преходите $\boldsymbol{\delta} : (\boldsymbol{Q}_1 \times \boldsymbol{Q}_2) \times \boldsymbol{\Sigma} \to \boldsymbol{Q}_1 \times \boldsymbol{Q}_2$ на новия автомат е дефинирана като $\boldsymbol{\delta}(\langle p_i, r_j\rangle, x) \stackrel{\text{деф}}{=} \langle \boldsymbol{\delta}_1(p_i, x), \boldsymbol{\delta}_2(r_j, x)\rangle$. Така получаваме автомата на *Фигура 3.7в*. Ако означим състоянията $\langle p_i, r_j\rangle$ като q_{ij}, то ще получим точно автомата от *Задача 3.2*. □

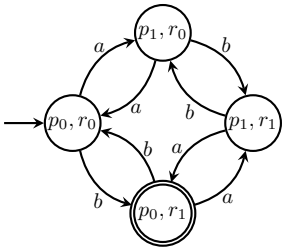
Фигура 3.7: Прилагаме декартова конструкция за да получим автомат за сечението на езиците на $\mathcal{A}_1$ и $\mathcal{A}_2$.



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



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



(в) $\mathcal{L}(\mathcal{A}) = L$

3.3 Каноничен автомат

В този раздел ще видим, че езиците могат да играят ролята на състояния на един автомат, който ние наричаме **автомат на Бжозовски** ([20, стр. 112], [8, стр. 53]). Тази конструкция ще бъде важна за нас и затова често ще наричаме автоматът на Бжозовски **каноничния автомат** за дадения език.

И така, нека е даден произволен език L. Ще покажем конструкция на детерминиран автомат

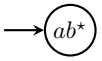
$$\mathcal{B}_L = \langle \boldsymbol{\Sigma}, \boldsymbol{Q}_L, \boldsymbol{s}_L, \boldsymbol{\delta}_L, \boldsymbol{F}_L \rangle,$$

който разпознава L. Този автомат ще наречем автомат на Бжозовски. Ще видим, 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:

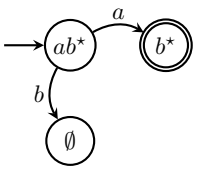
- Състоянията $\boldsymbol{Q}_L$ на автомата на Бжозовски ще бъдат езици над азбуката $\boldsymbol{\Sigma}$, където:
$$\boldsymbol{Q}_L \stackrel{\text{деф}}{=} \{\alpha^{-1}(L) \,:\, \alpha \in \boldsymbol{\Sigma}^\star\}.$$
Това е същото множество, което се разглежда в Раздел 2.6.
- Началното състояние $\boldsymbol{s}_L$ на автомата на Бжозовски е дефинирано като:
$$\boldsymbol{s}_L \stackrel{\text{деф}}{=} L \qquad // \; L = \varepsilon^{-1}(L).$$
- За произволно състояние (език) $M \in \boldsymbol{Q}_L$ и буква a, дефинираме функцията на преходите ето така:
$$\boldsymbol{\delta}_L(M, a) \stackrel{\text{деф}}{=} a^{-1}(M).$$
- Финалните състояния на автомата на Бжозовски са следните:
$$\boldsymbol{F}_L \stackrel{\text{деф}}{=} \{M \in \boldsymbol{Q}_L \,:\, \varepsilon \in M\}.$$

Фигура 3.8: Да разгледаме внимателно как се строи каноничния автомат за езика описван с регулярния израз $\boldsymbol{ab^\star}$. Стъпките са следните:

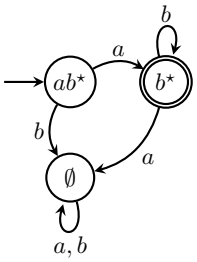
(а) Започваме с началното състояние $ab^\star$.



(б) След това получаваме две нови състояния, защото $a^{-1}(ab^\star) = b^\star$ и $b^{-1}(ab^\star) = \emptyset$. $b^\star$ е финално състояние, защото $\varepsilon \in b^\star$.



(в) Не получаваме нови състояния, защото $a^{-1}(b^\star) = \emptyset$ и $b^{-1}(b^\star) = b^\star$. Ясно е също, че $a^{-1}(\emptyset) = \emptyset$ и $b^{-1}(\emptyset) = \emptyset$.



Твърдение 3.5. За всяка дума ω и всяко състояние $M \in \boldsymbol{Q}_L$ е изпълнено равеството:

$$\boldsymbol{\delta}^\star_L(M, \omega) = \omega^{-1}(M). \tag{3.4}$$

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

✍ Опитайте се да докажете това твърдение сами!

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

□

Доказателство.

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

$$\boldsymbol{\delta}^\star_L(M, \varepsilon) \overset{\text{деф}}{=} M = \varepsilon^{-1}(M).$$

- Да примем, че за думи ω с дължина n е изпълнено Свойство (3.4).
- Сега да разгледаме дума ω с дължина $n+1$, т.е. $\omega = \beta a$ и $|\beta| = n$. Тогава

$$\begin{aligned}
\boldsymbol{\delta}^\star_L(M, \beta a) &= \boldsymbol{\delta}_L(\boldsymbol{\delta}^\star_L(M, \beta), a) && \text{// от деф. на } \boldsymbol{\delta}^\star_L \\
&= \boldsymbol{\delta}_L(\beta^{-1}(M), a) && \text{// от } \textbf{(И.П.)} \\
&= a^{-1}(\beta^{-1}(M)) && \text{// от деф. на } \boldsymbol{\delta}_L \\
&= (\beta a)^{-1}(M).
\end{aligned}$$

□

Твърдение 3.6. За произволен език L е изпълнено равенството $L = \mathcal{L}(\mathcal{B}_L)$.

Упътване. Съобразете, че имаме следните еквивалентности:

Тук е важно да отбележим, че все още не знаем дали $\mathcal{B}_L$ има крайно много състояния. В *Пример 3.1* ще видим един детерминиран безкраен автомат за език, който не е регулярен.

$$\begin{aligned}
\omega \in L &\Leftrightarrow \varepsilon \in \omega^{-1}(L) \\
&\Leftrightarrow \varepsilon \in \boldsymbol{\delta}^\star_L(L, \omega) && \text{// от } \textit{Твърдение 3.5} \\
&\Leftrightarrow \boldsymbol{\delta}^\star_L(\boldsymbol{s}_L, \omega) \in \boldsymbol{F}_L && \text{// } \varepsilon \in \omega^{-1}(L) \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{B}_L). && \text{// деф. на език на автомат}
\end{aligned}$$

□

Примери

Задача 3.5. Постройте каноничен автомат за $L \stackrel{\text{деф}}{=} \mathcal{L}(((\boldsymbol{a}+\boldsymbol{b})^+ \cdot \boldsymbol{a})^\star)$.

Решение.

(а) Веднага се вижда, че L ще бъде и финално състояние, защото $\varepsilon \in L$. Удобно е да имаме предвид следното представяне, при което явно да се вижда кои думи на L започват с буквата a и кои с b:

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

(б) 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 $\varepsilon \notin M$. Тогава имаме ново състояние M и

$$\begin{aligned} \boldsymbol{\delta}_L(L,a) &\stackrel{\text{деф}}{=} M && \text{// защото } a^{-1}(L) = M \\ \boldsymbol{\delta}_L(L,b) &\stackrel{\text{деф}}{=} M. && \text{// защото } b^{-1}(L) = M \end{aligned}$$

(в) За следващата стъпка е удобно да представим езика M по следния начин:

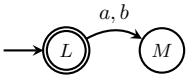
$$\begin{aligned} M &= \{a,b\}^\star aL \\ &= aL \cup \{a,b\}^+ aL \\ &= aL \cup \{a,b\} \cdot \{a,b\}^\star aL \\ &= aL \cup \{a,b\} \cdot M \\ &= aL \cup aM \cup bM \end{aligned}$$

От това представяне на M веднага се съобразява, че $a^{-1}(M) = L \cup M$ и $b^{-1}(M) = M$. 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

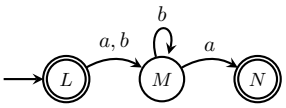
Фигура 3.9: Стъпките при строене на каноничния автомат за L:

(а) Започваме с началното състояние, което съответства на целия език L. То е и финално, защото $\varepsilon \in L$.

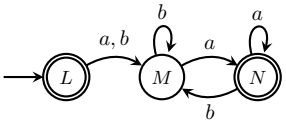
(б) Част от автомата, която показва преходите между състоянията L и M, като M не е финално, защото $\varepsilon \notin M$.



(в) Добавяме и състоянието N.



(г) Окончателно получаваме автомата на Бжозовски за езика L.



ново състояние N и тогава

$$\begin{aligned} \boldsymbol{\delta}_L(M,a) &\stackrel{\text{деф}}{=} N && \text{// защото } a^{-1}(M) = N \\ \boldsymbol{\delta}_L(M,b) &\stackrel{\text{деф}}{=} M && \text{// защото } b^{-1}(M) = M. \end{aligned}$$

(г) 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 $a^{-1}(N) = N$ и $b^{-1}(N) = M$. Сега полагаме:

$$\begin{aligned} \boldsymbol{\delta}_L(N,a) &\stackrel{\text{деф}}{=} N && \text{// защото } a^{-1}(N) = N \\ \boldsymbol{\delta}_L(N,b) &\stackrel{\text{деф}}{=} M && \text{// защото } b^{-1}(N) = M. \end{aligned}$$

Това означава, че нямаме повече нови състояния. Следователно, състоянията на автомата на Бжозовски са $\boldsymbol{Q}_L \stackrel{\text{деф}}{=} \{L, M, N\}$. Понеже $\varepsilon \in L$ и $\varepsilon \in N$ е ясно, че $\boldsymbol{F}_L = \{L, N\}$.

□

Задача 3.6. Постройте каноничен автомат за $L \stackrel{\text{деф}}{=} \mathcal{L}(\boldsymbol{a} \cdot (\boldsymbol{a} + \boldsymbol{b})^\star \cdot \boldsymbol{b})$.

Решение.

(а) Започваме с началното състояние L, за което е ясно, че няма да бъде финално, защото $\varepsilon \notin L$.

(б) Сега да видим производните на L относно a и b. Получаваме, че $a^{-1}(L) = \{a,b\}^\star b \stackrel{\text{деф}}{=} M$. Имаме, че $M \neq L$, защото $b \in M$, но $b \notin L$. Тогава

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

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

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

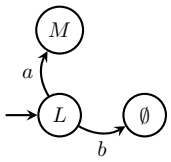
(г) Щом имам представянето $N = \{\varepsilon\} \cup aM \cup bM \cup \{b\}$, то е ясно, че $a^{-1}(N) = M$ и $b^{-1}(N) = M$. Тогава $\boldsymbol{\delta}_L(N,a) \stackrel{\text{деф}}{=} M$ и $\boldsymbol{\delta}_L(N,b) \stackrel{\text{деф}}{=} N$. Понеже $x^{-1}(\emptyset) = \emptyset$ за всяка буква x, имаме примки за състоянието $\emptyset$, т.е. $\boldsymbol{\delta}_L(\emptyset,a) \stackrel{\text{деф}}{=} \emptyset$ и $\boldsymbol{\delta}_L(\emptyset,b) \stackrel{\text{деф}}{=} \emptyset$. Съобразете сами, че $\boldsymbol{F}_L = \{N\}$.

□

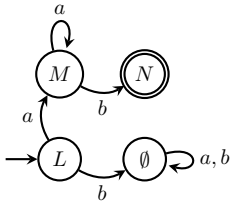
Фигура 3.10: Стъпките при строене на каноничния автомат за L:

(а) Започваме с началното състояние, което съответства на целия език L.

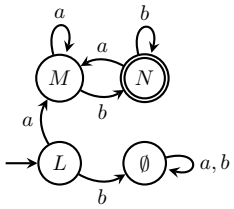
(б) Част от автомата със състояния съответстващи на езиците L, M и $\emptyset$.



(в) Добавяме състояние за езика N. То е финално, защото $\varepsilon \in N$.



(г) Каноничен автомат за L по метода на Бжозовски.



Задача 3.7. Постройте каноничен автомат за езика

$$L \stackrel{\text{деф}}{=} \{\omega \in \{a,b\}^\star \,:\, |\omega|_a \text{ е четно и } |\omega|_b = 1\}.$$

Решение. Нека да положим за $i, j < 2$,

$$L_{i,j} \stackrel{\text{деф}}{=} \{\omega \in \{a,b\}^\star \,:\, |\omega|_a \equiv i \bmod 2 \;\&\; |\omega|_b = j\}.$$

Ясно е, че $L = L_{0,1}$ и всички тези езици $L_{i,j}$ са различни. Да започнем с преходите от началното състояние за езика $L_{0,1}$:

$$a^{-1}(L_{0,1}) = L_{1,1} \text{ и } b^{-1}(L_{0,1}) = L_{0,0}$$

Състоянието за езика $L_{0,0}$ е финално, защото $\varepsilon \in L_{0,0}$. Това ще бъде и единственото финално състояние. Сега да видим какви преходи имаме от състоянието за езика $L_{1,1}$:

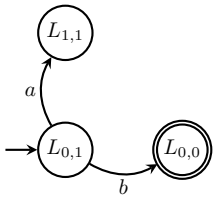
$$a^{-1}(L_{1,1}) = L_{0,1} \text{ и } b^{-1}(L_{1,1}) = L_{1,0}.$$

Продължаваме по същия начин за състоянията за останалите езици и така получаваме автомата на *Фигура 3.11*. □

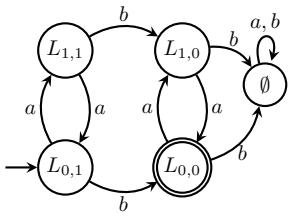
Фигура 3.11: Автомат, който приема думи с четен брой a и точно едно b, получен чрез метода на Бжозовски.

(а) Започваме с началното състояние, което съответства на целия език $L_{0,1}$.

(б) Добавяме нови състояния достижими от $L_{0,1}$.



(в) Продължаваме така докато получим целия автомат



Допълнителни задачи

Задача 3.8. Постройте каноничен автомат за $L \stackrel{\text{деф}}{=} \{\alpha \in \{0,1\}^\star : \overline{\alpha}_{(2)} \equiv 2 \bmod 3\}$.

Забележка. Да припомним, че в *Задача 3.3* се искаше да се докаже, че същият език е регулярен. Ние направихме това като построихме автомат за L и доказахме, че той разпознава L.

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:

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

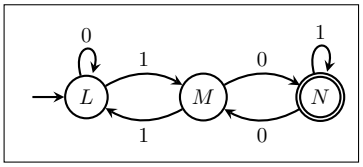
Лесно се съобразява, че езикът M е различен от L, защото например думата $10 \in L$, но $10 \notin M$. Също така, понеже $2^{|\varepsilon|} + \overline{\varepsilon}_{(2)} = 1$, то $\varepsilon \notin M$ и следователно M няма да бъде финално състояние в автомата на Бжозовски. Продължаваме нататък:

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

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

Сега трябва да се уверим, че езикът N е различен от L и M. Непосредствено се проверява, че думата $11 \in N$, но $11 \notin L$ и $11 \notin M$. 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:

$$\begin{aligned}
0^{-1}(N) &= \{\alpha : 0\alpha \in N\} \\
&= \{\alpha : 2 \cdot 2^{|0\alpha|} + \overline{0\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha : 4 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha : 3 \cdot 2^{|\alpha|} + 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha : 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= M \\
1^{-1}(N) &= \{\alpha : 1\alpha \in N\} \\
&= \{\alpha : 2 \cdot 2^{|1\alpha|} + \overline{1\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha : 4 \cdot 2^{|\alpha|} + 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha : 3 \cdot 2^{|\alpha|} + 2 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= \{\alpha : 2 \cdot 2^{|\alpha|} + \overline{\alpha}_{(2)} \equiv 2 \bmod 3\} \\
&= N.
\end{aligned}$$



Фигура 3.12: Автомат на Бжозовски за езика L.

□

Пример 3.1. Да разгледаме отново езика

$$L \stackrel{\text{деф}}{=} \{a^n b^n \ : \ n \in \mathbb{N}\}.$$

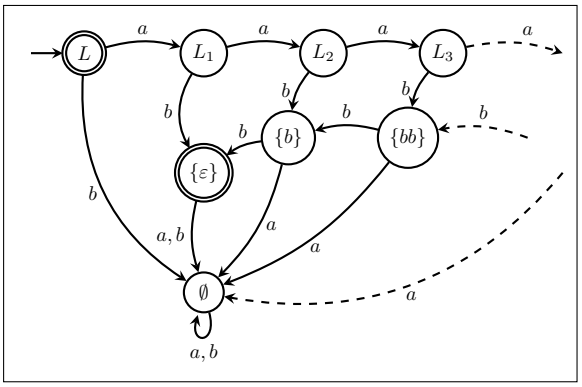
Ние знаем от *Пример 3.5*, че L не е регулярен език. Да се опитаме да построим автомат, който го разпознава. Нека за всяко k да разгледаме езика

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

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*. Този пример се разглежда и в [20, стр. 113].



Фигура 3.13: *Безкраен* автомат за $L = \{a^n b^n \ : \ n \in \mathbb{N}\}$ построен по метода на Бжозовски.

3.4 Регулярните езици са автоматни

Нашата цел в този раздел ще бъде да покажем как като използваме директно конструкцията на Бжозовски можем да покажем, че автоматните езици са затворени относно операциите обединение, конкатенация и звезда на Клини. За щастие, ние вече сме свършили тежката работа в *Теорема 2.2*, която гласи, че ако езикът L е регулярен, то $\boldsymbol{Q}_L$ е крайно множество. Понеже елементите на $\boldsymbol{Q}_L$ играят ролята на състояния на автомата $\mathcal{B}_L$ за езика L, то по този начин веднага получаваме следната теорема.

Теорема 3.1 (Клини[12])**.** Всеки регулярен език е автоматен.

Алтернативно доказателство на тази теорема е дадено в *Раздел 3.11*.

Доказателство. Знаем от *Твърдение 3.6*, че $\mathcal{L}(\mathcal{B}_L) = L$. Щом L е регулярен език, то от *Теорема 2.2* имаме, че $\boldsymbol{Q}_L$ е крайно множество и следователно $\mathcal{B}_L$ е ДКА. □

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

Твърдение 3.7. Ако L_1 и L_2 са регулярни езици, то $L_1 \cap L_2$ и $L_1 \setminus L_2$ са регулярни езици.

Ние знаем, че автоматните езици са затврени относно сечение и разлика чрез декартови конструкции на автомати.
🔍 Сравнете с *Твърдение 3.2*, където прилагаме автоматен подход за доказателство на тези свойства.

Упътване. Нека L_1 и L_2 са регулярни езици. Тогава от *Теорема 2.2* следва, че $\boldsymbol{Q}_{L_1}$ и $\boldsymbol{Q}_{L_2}$ са крайни. Знаем, че $\omega^{-1}(L_1 \cap L_2) = \omega^{-1}(L_1) \cap \omega^{-1}(L_2)$. Това означава, че

$$\boldsymbol{Q}_{L_1 \cap L_2} \subseteq \{M_1 \cap M_2 \,:\, M_1 \in \boldsymbol{Q}_{L_1} \,\&\, M_2 \in \boldsymbol{Q}_{L_2}\}.$$

Следователно, веднага можем да получим следната горна граница за броя на елементите на $\boldsymbol{Q}_{L_1 \cap L_2}$:

$$|\boldsymbol{Q}_{L_1 \cap L_2}| \leq |\boldsymbol{Q}_{L_1} \times \boldsymbol{Q}_{L_2}| = |\boldsymbol{Q}_{L_1}| \cdot |\boldsymbol{Q}_{L_2}|.$$

Щом $\boldsymbol{Q}_{L_1 \cap L_2}$ е крайно, то можем да построим каноничен автомат за $L_1 \cap L_2$, което означава, че $L_1 \cap L_2$ е автоматен език и оттук $L_1 \cap L_2$ е регулярен. Разсъждаваме аналогично и за $L_1 \setminus L_2$. □

3.5 Изоморфни автомати

За два детерминирани крайни автомати $\mathcal{A}_1$ и $\mathcal{A}_2$ казваме, че $\mathcal{A}_1$ и $\mathcal{A}_2$ са **изоморфни**, ако съществува *биективна* функция $f : \boldsymbol{Q}_1 \to \boldsymbol{Q}_2$, за която:

(1) $f(\boldsymbol{s}_1) = \boldsymbol{s}_2$;

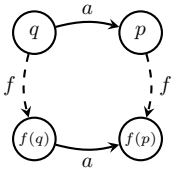
(2) $q \in \boldsymbol{F}_1 \Leftrightarrow f(q) \in \boldsymbol{F}_2$;

(3) $f(\boldsymbol{\delta}_1(q, a)) = \boldsymbol{\delta}_2(f(q), a)$.

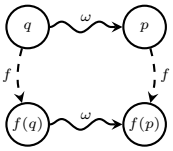
С други думи, два автомата са изоморфни точно тогава, когато те са идентични с точност до преименуване на състоянията. Ще казваме, че f задава изоморфизъм на $\mathcal{A}_1$ върху $\mathcal{A}_2$ и ще означаваме $f : \mathcal{A}_1 \cong \mathcal{A}_2$.

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

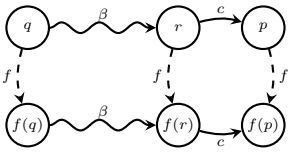
Условие (3) може да се представи графично така:



Свойство 3.5 може да се представи графично така:



Индукционната стъпка може да се представи така:



Твърдение 3.8. Нека $f : \mathcal{A}_1 \cong \mathcal{A}_2$. Тогава за всяка дума ω и състояние $q \in \boldsymbol{Q}_1$ е изпълнено:

$$f(\boldsymbol{\delta}_1^\star(q, \omega)) = \boldsymbol{\delta}_2^\star(f(q), \omega). \tag{3.5}$$

Доказателство. Ще докажем *Свойство 3.5* с индукция по дължината на ω.

- Нека $|\omega| = 0$, т.е. $\omega = \varepsilon$. Ясно е, че *Свойство 3.5* е изпълнено за ε защото $\boldsymbol{\delta}_1^\star(q, \varepsilon) = q$ и съответно $\boldsymbol{\delta}_2^\star(f(q), \varepsilon) = f(q)$ за произволно $q \in \boldsymbol{Q}_1$.
- Да приемем, че *Свойство 3.5* е изпълнено за думи с дължина n.
- Да разгледаме произволна дума ω с дължина $n+1$, т.е. $\omega = \beta c$ и $|\beta| = n$. Тогава имаме следната верига от равенства:

$$\begin{aligned}
f(\boldsymbol{\delta}_1^\star(q, \beta c)) &= f(\boldsymbol{\delta}_1(\overbrace{\boldsymbol{\delta}_1^\star(q, \beta)}^{p}, c)) \\
&= f(\boldsymbol{\delta}_1(p, c)) && \text{// } \textit{Свойство (3)} \\
&= \boldsymbol{\delta}_2(f(p), c) \\
&= \boldsymbol{\delta}_2(f(\boldsymbol{\delta}_1^\star(q, \beta)), c)) \\
&= \boldsymbol{\delta}_2(\boldsymbol{\delta}_2^\star(f(q), \beta), c) && \text{// } \textbf{(И.П.)} \text{ за } \beta \\
&= \boldsymbol{\delta}_2^\star(f(q), \beta c).
\end{aligned}$$

□

Твърдение 3.9. 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 $f : \mathcal{A}_1 \cong \mathcal{A}_2$. Тогава имаме следните еквивалентности:

$$\begin{aligned}
\omega \in \mathcal{L}(\mathcal{A}_1) &\Leftrightarrow \boldsymbol{\delta}_1^\star(\boldsymbol{s}_1, \omega) \in \boldsymbol{F}_1 && \text{// деф. на } \mathcal{L}(\mathcal{A}_1) \\
&\Leftrightarrow f(\boldsymbol{\delta}_1^\star(\boldsymbol{s}_1, \omega)) \in \boldsymbol{F}_2 && \text{// } \textit{Свойство (2)} \\
&\Leftrightarrow \boldsymbol{\delta}_2^\star(f(\boldsymbol{s}_1), \omega)) \in \boldsymbol{F}_2 && \text{// } \textit{Твърдение 3.8} \\
&\Leftrightarrow \boldsymbol{\delta}_2^\star(\boldsymbol{s}_2, \omega) \in \boldsymbol{F}_2 && \text{// } f(\boldsymbol{s}_1) \overset{\text{деф}}{=} \boldsymbol{s}_2 \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{A}_2). && \text{// деф. на } \mathcal{L}(\mathcal{A}_2)
\end{aligned}$$

□

3.6 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 $q_\alpha \stackrel{\text{деф}}{=} \boldsymbol{\delta}^\star(\boldsymbol{s}, \alpha)$. 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 $\boldsymbol{\delta}$ е тотална функция. За някои състояния p може да съществуват безкрайно много думи α, за които $q_\alpha = p$.

Твърдение 3.10. 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:

$$\begin{aligned}
\omega \in \mathcal{L}_\mathcal{A}(q_\alpha) &\Leftrightarrow \boldsymbol{\delta}^\star(q_\alpha, \omega) \in \boldsymbol{F} && \text{// от деф. на } \mathcal{L}_\mathcal{A}(q_\alpha) \\
&\Leftrightarrow \boldsymbol{\delta}^\star(\boldsymbol{\delta}^\star(\boldsymbol{s}, \alpha), \omega) \in \boldsymbol{F} && \text{// } q_\alpha \stackrel{\text{деф}}{=} \boldsymbol{\delta}^\star(\boldsymbol{s}, \alpha) \\
&\Leftrightarrow \boldsymbol{\delta}^\star(\boldsymbol{s}, \alpha\omega) \in \boldsymbol{F} && \text{// } \textit{Твърдение 3.1} \\
&\Leftrightarrow \alpha\omega \in \mathcal{L}(\mathcal{A}) && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\
&\Leftrightarrow \alpha\omega \in L && \text{// } L = \mathcal{L}(\mathcal{A}) \\
&\Leftrightarrow \omega \in \alpha^{-1}(L). && \text{// деф. на } \alpha^{-1}(L)
\end{aligned}$$

□

Твърдение 3.11. 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 ω имаме

$$\begin{aligned} \omega \in \mathcal{L}_{\mathcal{B}_L}(M) &\Leftrightarrow \boldsymbol{\delta}^\star_L(M,\omega) \in \boldsymbol{F}_L \\ &\Leftrightarrow \varepsilon \in \omega^{-1}(M) && \text{// деф. на } \boldsymbol{F}_L \\ &\Leftrightarrow \omega \in M, \end{aligned}$$

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

Лема 3.1. Нека L е регулярен език и $\mathcal{A}$ е произволен ДКА, за който $L = \mathcal{L}(\mathcal{A})$. Тогава $|\boldsymbol{Q}_L| \leq |\boldsymbol{Q}_\mathcal{A}|$.

Тази лема ни казва, че автоматът на Бжозовски има възможно най-малкия брой състояния измежду всички ДКА разпознаващи L [20, стр. 113].

Доказателство. Да разгледаме функцията $f : \boldsymbol{Q}_\mathcal{A} \to \boldsymbol{Q}_L$ зададена по следния начин:

$$f(q) \stackrel{\text{деф}}{=} \mathcal{L}_\mathcal{A}(q). \tag{3.6}$$

Първо, да видим защо f е добре дефинирана функция, т.е. да видим защо за всяко състояние $q \in \boldsymbol{Q}_\mathcal{A}$, имаме $f(q) \in \boldsymbol{Q}_L$. 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 $p = \boldsymbol{\delta}^\star_\mathcal{A}(\boldsymbol{s}_\mathcal{A}, \alpha)$, т.е. $p = q_\alpha$ според означението, което въведохме в началото на *Раздел 3.6*. Тогава от *Твърдение 3.10* следва, че $f(q_\alpha) = \alpha^{-1}(L) \in \boldsymbol{Q}_L$, 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 $M \in \boldsymbol{Q}_L$, което означава, че има дума α, за която $M = \alpha^{-1}(L)$. Отново според *Твърдение 3.10*,

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

Сега от *Твърдение 1.1* можем да заключим, че $|\boldsymbol{Q}_L| \leq |\boldsymbol{Q}_\mathcal{A}|$. □

Следствие 3.1 (Критерий за регулярност на език)**.** Един език L е регулярен точно тогава, когато $\boldsymbol{Q}_L$ е крайно множество.

Доказателство. Първо, ако L е регулярен, то от *Теорема 2.2* веднага имаме, че $\boldsymbol{Q}_L$ е крайно. Второ, ако $\boldsymbol{Q}_L$ е крайно, то $\mathcal{B}_L$ е ДКА. Понеже $\mathcal{L}(\mathcal{B}_L) = L$ според *Твърдение 3.6*, то L е автоматен и следователно регулярен по теоремата на Клини, за която имаме доказателство с алгоритмичен подход и алгебричен подход. □

Следствие 3.2. 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.

Сега ще видим, че всеки два детерминирани крайни автомати за даден език L с минимален възможен брой състояния са на практика едни и същи - те съвпадат с точност до преимуновване на състоянията.

✍ Проверете, че за произволен регулярен език L, за всеки ДКА $\mathcal{A}$, за който $\mathcal{L}(\mathcal{A}) = L$ е изпълнено, че $|\boldsymbol{F}_L| \leq |\boldsymbol{F}_\mathcal{A}|$.

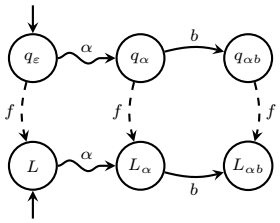
Теорема 3.2. За всеки регулярен език L съществува *единствен* минимален ДКА с точност до изоморфизъм.

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 $|\boldsymbol{Q}_\mathcal{A}| = |\boldsymbol{Q}_L|$. 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 $f : \boldsymbol{Q}_\mathcal{A} \to \boldsymbol{Q}_L$, където

$$f(p) \overset{\text{деф}}{=} \mathcal{L}_\mathcal{A}(p). \tag{3.7}$$

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

От *Лема 3.1* знаем, че f е сюрективна. Понеже $|\boldsymbol{Q}_\mathcal{A}| = |\boldsymbol{Q}_L|$, то от *Твърдение 1.2* имаме, че f е всъщност биекция. Остава да видим защо $\mathcal{A} \cong_f \mathcal{B}_L$. Да напомним, че можем да разглеждаме състоянията на $\boldsymbol{Q}_\mathcal{A}$ като q_α и тогава $\mathcal{L}_\mathcal{A}(q_\alpha) = \alpha^{-1}(L)$ от *Твърдение 3.10*.



Тук сме означили

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

- За началното състояние имаме, че:

$$\begin{aligned} f(\boldsymbol{s}_\mathcal{A}) &= f(q_\varepsilon) && \text{// } \boldsymbol{s}_\mathcal{A} = q_\varepsilon \\ &= \mathcal{L}_\mathcal{A}(q_\varepsilon) && \text{// от деф. на } f \\ &= \varepsilon^{-1}(L) && \text{// } \textit{Твърдение 3.10} \\ &= L. && \text{// } \boldsymbol{s}_L = L \end{aligned}$$

- За финалните състояния имаме, че:

$$\begin{aligned} q_\alpha \in \boldsymbol{F}_\mathcal{A} &\Leftrightarrow \boldsymbol{\delta}^\star_\mathcal{A}(\boldsymbol{s}, \alpha) \in \boldsymbol{F}_\mathcal{A} && \text{// } q_\alpha = \boldsymbol{\delta}^\star_\mathcal{A}(\boldsymbol{s}, \alpha) \\ &\Leftrightarrow \alpha \in \mathcal{L}(\mathcal{A}) && \text{// деф. на } \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \varepsilon \in \alpha^{-1}(L) && \text{// } L = \mathcal{L}(\mathcal{A}) \\ &\Leftrightarrow \alpha^{-1}(L) \in \boldsymbol{F}_L. && \text{// деф. на } \boldsymbol{F}_L \end{aligned}$$

- Остава да докажем, че за произволна буква b и произволни състояния $q, p \in$

$\boldsymbol{Q}_{\mathcal{A}}$ е изпълнено, че $\boldsymbol{\delta}_L(f(q), b) = f(\boldsymbol{\delta}_{\mathcal{A}}(q, b))$, или с други думи,

$$\boldsymbol{\delta}_L(\mathcal{L}_{\mathcal{A}}(q), b) = \mathcal{L}_{\mathcal{A}}(\boldsymbol{\delta}_{\mathcal{A}}(q, b)). \tag{3.8}$$

Това се вижда директно от веригата от еквивалентности:

$$\begin{aligned} \omega \in \boldsymbol{\delta}_L(\mathcal{L}_{\mathcal{A}}(q), b) &\Leftrightarrow \omega \in b^{-1}(\mathcal{L}_{\mathcal{A}}(q)) \\ &\Leftrightarrow b\omega \in \mathcal{L}_{\mathcal{A}}(q) \\ &\Leftrightarrow \boldsymbol{\delta}^{\star}_{\mathcal{A}}(\boldsymbol{\delta}_{\mathcal{A}}(q, b), \omega) \in \boldsymbol{F}_{\mathcal{A}} \\ &\Leftrightarrow \omega \in \mathcal{L}_{\mathcal{A}}(\boldsymbol{\delta}_{\mathcal{A}}(q, b)). \end{aligned}$$

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

Следствие 3.3 (Критерий за минималност). Нека $\mathcal{A}$ е ДКА, при който всяко състояние е достижимо от началното, разпознаващ регулярния език L. Тогава $\mathcal{A}$ е *минимален* автомат за езика L точно тогава, когато е изпълнена импликацията:

$$(\forall p \in \boldsymbol{Q}_{\mathcal{A}})(\forall q \in \boldsymbol{Q}_{\mathcal{A}})[\, p \neq q \implies \mathcal{L}_{\mathcal{A}}(p) \neq \mathcal{L}_{\mathcal{A}}(q)\,]. \tag{3.9}$$

Биекция = инекция + сюрекция.

Доказателство. Нека $\mathcal{B}_L$ е автоматът на Бжозовски за L и да разгледаме функцията $f : \boldsymbol{Q}_{\mathcal{A}} \to \boldsymbol{Q}_L$ дефинирана като $f(q) = \mathcal{L}_{\mathcal{A}}(q)$. Първо, нека $\mathcal{A}$ е минимален автомат за L. Тогава знаем от *Теорема 3.2*, че f е биекция. От инективността на f следва, че имаме импликцията (3.9).

Второ, нека импликацията (3.9) е изпълнена. Това означава, че f е инективна функция, откъдето имаме, че $|\boldsymbol{Q}_{\mathcal{A}}| \leq |\boldsymbol{Q}_L|$. Понеже $\mathcal{B}_L$ е минимален автомат, то $\mathcal{A}$ също е минимален автомат. □

3.7 Минимизация

Нека да започнем с едно твърдение, което представлява критерий, кога детерминизацията ни дава автомат с минимален брой състояния.

Тук ще изложим алгебричен подход за намиране на минимален автомат. Този подход е описан добре в [21, стр. 88].

Твърдение 3.12. Нека $\mathcal{A}$ е НКА със следните свойства:

(а) $(\forall q \in \boldsymbol{Q}_{\mathcal{A}})[\, \mathcal{L}_{\mathcal{A}}(q) \neq \emptyset \,]$;

(б) $(\forall p \in \boldsymbol{Q}_{\mathcal{A}})(\forall q \in \boldsymbol{Q}_{\mathcal{A}})[\, p \neq q \implies \mathcal{L}_{\mathcal{A}}(q) \cap \mathcal{L}_{\mathcal{A}}(p) = \emptyset \,]$.

Тогава $\mathcal{M} = \mathtt{det}(\mathcal{A})$, получен по метода на Рабин-Скот, е *минимален* автомат за езика на $\mathcal{A}$.

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:

Критерият за минималност на детерминиран автомат $\mathcal{A}$ 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 (б), 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.

$$\mathcal{L}_{\mathcal{M}}(P) = \mathcal{L}_{\mathcal{M}}(U) \implies P = U.$$

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 (а) имаме, че $\mathcal{L}_{\mathcal{A}}(p) \neq \emptyset$, то да разгледаме една дума $\omega \in \mathcal{L}_{\mathcal{A}}(p)$. Знаем, че $\boldsymbol{\Delta}^{\star}_{\mathcal{A}}(\{p\}, \omega) \cap \boldsymbol{F}_{\mathcal{A}} \neq \emptyset$ или с други думи, $\boldsymbol{\delta}^{\star}_{\mathcal{M}}(P, \omega) \in \boldsymbol{F}_{\mathcal{M}}$, 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 $\boldsymbol{\delta}^{\star}_{\mathcal{M}}(U, \omega) \in \boldsymbol{F}_{\mathcal{M}}$, което означава, че за някое състояние $u \in U$, $\boldsymbol{\Delta}^{\star}_{\mathcal{A}}(\{u\}, \omega) \cap \boldsymbol{F}_{\mathcal{A}} \neq \emptyset$. Така получаваме, че $\omega \in \mathcal{L}_{\mathcal{A}}(u)$. От (б) следва, че $u = p$, защото $\mathcal{L}_{\mathcal{A}}(p) \cap \mathcal{L}_{\mathcal{A}}(u) \neq \emptyset$. Заключаваме, че $P \subseteq U$. □

Твърдение 3.13. Нека $\mathcal{D}$ е ДКА, за който всяко състояние е достижимо от началното. Да разгледаме $\mathcal{A} \stackrel{\text{деф}}{=} \mathtt{rev}(\mathcal{D})$. Тогава са изпълнени двете свойства от условието на *Твърдение 3.12*, а именно:

(а) $(\forall q \in \boldsymbol{Q}_{\mathcal{A}})[\, \mathcal{L}_{\mathcal{A}}(q) \neq \emptyset \,]$;

(б) $(\forall p \in \boldsymbol{Q}_{\mathcal{A}})(\forall q \in \boldsymbol{Q}_{\mathcal{A}})[\, p \neq q \implies \mathcal{L}_{\mathcal{A}}(q) \cap \mathcal{L}_{\mathcal{A}}(p) = \emptyset \,]$.

Упътване. Да напомним, че според *Задача 3.30*, $\mathcal{A} = (\boldsymbol{\Sigma}, \boldsymbol{Q}, \boldsymbol{\Delta}, \boldsymbol{F}, \{\boldsymbol{s}\})$, където

$$\boldsymbol{\Delta}_{\mathcal{A}}(q, a) \stackrel{\text{деф}}{=} \{p \in \boldsymbol{Q} \,:\, \boldsymbol{\delta}_{\mathcal{D}}(p, a) = q\}.$$

Щом в $\mathcal{D}$ всяко състояние q е достижимо от началното, а в $\mathcal{A}$ състоянието s е финално, 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 $\mathbf{\Delta}^{\star}_{\mathcal{A}}(\{p\}, \omega) \ni s$, защото s е единственото финално състояние на $\mathcal{A}$. Това означава, че $\delta^{\star}_{\mathcal{D}}(s, \omega) = p$. От друга страна, имаме също, че $\omega \in \mathcal{L}_{\mathcal{A}}(q)$, откъдето получаваме, че $\delta^{\star}_{\mathcal{D}}(s, \omega) = q$. Ясно е, че $p = q$, защото $\mathcal{D}$ е детерминиран автомат и с една дума ω можем да отидем само в едно състояние като тръгнем от s. □

Лема 3.2. Нека $\mathcal{D}$ е ДКА, за който всяко състояние е достижимо от началното. Тогава детерминираният краен автомат $\mathcal{M}$ получен като

$$\mathcal{M} \stackrel{\text{деф}}{=} \mathtt{det}(\mathtt{rev}(\mathcal{D}))$$

е минимален за езика $\mathtt{Rev}(L(\mathcal{D}))$.

Упътване. Просто комбинираме горните две твърдения. Да започнем с $\mathcal{A} \stackrel{\text{деф}}{=} \mathtt{rev}(\mathcal{D})$. Знаем, че $\mathcal{L}(\mathcal{A}) = \mathtt{Rev}(\mathcal{L}(\mathcal{D}))$. От *Твърдение 3.13* знаем, че $\mathcal{A}$ притежава свойствата необходими за приложението на *Твърдение 3.12*. Така получаваме, че $\mathcal{M} \stackrel{\text{деф}}{=} \mathtt{det}(\mathcal{A})$ е минимален детерминиран автомат за езика $\mathcal{L}(\mathcal{A})$. □

По този начин получаваме алгоритъм за минимизация на автомат.

Теорема 3.3 (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 $\mathcal{A}$.

3.8 Недетерминирани крайни автомати

Въведени от Рабин и Скот [18]. За поголяма яснота, често ще означаваме с $\mathcal{N}$ недетерминирани автомати, като ще запазим $\mathcal{A}$ за детерминираните. Мотивация - [13, стр. 25].

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. Това е пример **недетерминизъм** - важно понятие в информатиката. Най-общо казано, недетерминизмът настъпва, когато попаднем в ситуация, при която следващата стъпка в изчислението не е еднозначно определено от текущото състояние. В нашия конкретен случай - текущото състояние е просто един елемент на множеството $\boldsymbol{Q}$. При по-сложни модели, текущото състоянието ще бъде по-сложен обект. Това няма да бъде единствения ни сблъсък с този феномен.

Определение 3.3. Недетерминиран краен автомат представлява петорка от вида

$$\mathcal{A} = \langle \boldsymbol{\Sigma}, \boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{\Delta}, \boldsymbol{F} \rangle,$$

където

- $\boldsymbol{Q}$ е крайно множество от състояния;
- $\boldsymbol{\Sigma}$ е крайна азбука;
- $\boldsymbol{\Delta} : \boldsymbol{Q} \times \boldsymbol{\Sigma} \to \mathscr{P}(\boldsymbol{Q})$ 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 $\boldsymbol{\Delta}(q, a) = \emptyset$;
- $\boldsymbol{S} \subseteq \boldsymbol{Q}$ е множество от начални състояния;
- $\boldsymbol{F} \subseteq \boldsymbol{Q}$ е множеството от финални състояния.

Да напомним означението $\mathscr{P}(\boldsymbol{Q}) \stackrel{\text{деф}}{=} \{R : R \subseteq \boldsymbol{Q}\}$ и свойството $|\mathscr{P}(\boldsymbol{Q})| = 2^{|\boldsymbol{Q}|}$. Няма единен стандарт за дефиниция на недетерминиран краен автомат.

- В [15] Δ е релация и се позволяват ε-преходи.
- В [22] пък е функция, но пак се позволяват ε-преходи.
- В [10] е функция без ε-преходи. Навсякъде има само едно начално.

Удобно е да разширим функцията на преходите $\boldsymbol{\Delta} : \boldsymbol{Q} \times \boldsymbol{\Sigma} \to \mathscr{P}(\boldsymbol{Q})$, която казва как правим преход с една буква от дадено състояние, до функцията $\boldsymbol{\Delta}^\star : \mathscr{P}(\boldsymbol{Q}) \times \boldsymbol{\Sigma}^\star \to \mathscr{P}(\boldsymbol{Q})$, която казва как правим преход с цяла дума от дадено множество от състояния.

Дефинираме $\boldsymbol{\Delta}^\star$ рекурсивно за произволно множество от състояния $R \subseteq \boldsymbol{Q}$ и дума $\alpha \in \boldsymbol{\Sigma}^\star$ по следния начин:

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

Понеже операцията $\bigcup$ е сложна, да напомним, че $\bigcup\{\{0,1\},\{2\},\{2,3\}\} = \{0,1\} \cup \{2\} \cup \{2,3\}$.

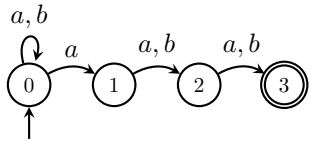
Дефиницията на език разпознаван от недерерминиран краен автомат $\mathcal{N}$ е малко по-сложна от тази за ДКА.

$$\mathcal{L}(\mathcal{N}) \stackrel{\text{деф}}{=} \{\omega \in \boldsymbol{\Sigma}^\star : \boldsymbol{\Delta}^\star(\boldsymbol{S}, \omega) \cap \boldsymbol{F} \neq \emptyset\}.$$

Удачно е да дефинираме език на автомата $\mathcal{N}$, ако приемем, че множеството от състояния R са начални. Това можем да направим така:

$$\mathcal{L}_{\mathcal{N}}(R) = \{\omega \in \mathbf{\Sigma}^\star \ : \ \mathbf{\Delta}^\star(R, \omega) \cap \boldsymbol{F} \neq \emptyset\}.$$

Фигура 3.15: Недетерминиран автомат $\mathcal{N}$ за езика съставен думите, в които има a на трета позиция от дясно на ляво.



Ако $R = \{q\}$, удобно е да пишем просто $\mathcal{L}_{\mathcal{N}}(q)$ вместо тромавия запис $\mathcal{L}_{\mathcal{N}}(\{q\})$. Отново, както при случая за детерминираните крайни автомати, удачно е да имаме възможност да разглеждаме прозиволно множество от състояния като финални. Да положим

$$\mathcal{L}_{\mathcal{N}}(R, P) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \ : \ \mathbf{\Delta}^\star(R, \omega) \cap P \neq \emptyset\}.$$

В частност, $\mathcal{L}(\mathcal{N}) = \mathcal{L}_{\mathcal{N}}(\boldsymbol{S}, \boldsymbol{F})$. Отново, в случая, когато $R = \{q\}$, ще пишем просто $\mathcal{L}_{\mathcal{N}}(q, P)$ вместо $\mathcal{L}_{\mathcal{N}}(\{q\}, P)$ и ако $P = \{p\}$, то можем да пишем просто $\mathcal{L}_{\mathcal{N}}(q, p)$.

Да разгледаме няколко примера как точно работи функцията $\mathbf{\Delta}^\star$ за автомата $\mathcal{N}$:

$$\mathbf{\Delta}^\star(\{0,1\}, ab) = \{0,2,3\},$$
$$\mathbf{\Delta}^\star(\{0,1\}, ba) = \{0,1,3\}.$$

Думата $babb \in \mathcal{L}(\mathcal{N})$, защото:

$$\begin{aligned}\mathbf{\Delta}^\star(\{0\}, babb) &= \mathbf{\Delta}^\star(\{0\}, abb)\\ &= \mathbf{\Delta}^\star(\{0,1\}, bb)\\ &= \mathbf{\Delta}^\star(\{0,2\}, b)\\ &= \{0,3\}.\end{aligned}$$

Да напомним, че в *Твърдение 3.1* доказахме следното свойството за ДКА:

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

Естествено е да проверим, че имаме същото свойство и за НКА.

Твърдение 3.14. За всеки две думи $\alpha, \beta \in \mathbf{\Sigma}^\star$ и всяко $R \subseteq \boldsymbol{Q}$,

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

Доказателство. Ще докажем, че $(\forall n \in \mathbb{N})P(n)$ с индукция по n, където

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

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

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

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

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

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

□

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

$$\frac{p \in \boldsymbol{\Delta}(q,a) \qquad \beta \in \boldsymbol{\Sigma}^\star}{(q, a\beta) \vdash_{\mathcal{N}} (p, \beta)}$$

Фигура 3.16: Едностъпков преход в недетерминиран краен автомат $\mathcal{N}$

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

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

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

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

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

Получаваме, че

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

$$\mathcal{L}(\mathcal{N}) = \{\alpha \in \boldsymbol{\Sigma}^\star \,:\, (\exists q \in \boldsymbol{S})(\exists f \in \boldsymbol{F})[(q,\alpha) \vdash^\star_{\mathcal{N}} (f, \varepsilon)]\}.$$

Теорема 3.4 (Рабин-Скот [18]). За всеки НКА $\mathcal{N}$ съществува еквивалентен на него ДКА $\mathcal{D}$, т.е.

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

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

$$\mathcal{D} = (\boldsymbol{\Sigma}, \boldsymbol{Q}', \boldsymbol{s}, \boldsymbol{\delta}, \boldsymbol{F}'),$$

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

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

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

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

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

- $\boldsymbol{s} \stackrel{\text{деф}}{=} \boldsymbol{S}$;
- $\boldsymbol{F}' \stackrel{\text{деф}}{=} \{R \in \boldsymbol{Q}' \,:\, R \cap \boldsymbol{F} \neq \emptyset\}$.

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

$$\boldsymbol{\Delta}^\star(\underbrace{R}_{\text{множество}}, \alpha) = \boldsymbol{\delta}^\star(\underbrace{R}_{\text{състояние}}, \alpha). \tag{3.10}$$

Това ще направим с индукция по дължината на думата α.

- Ако $|\alpha| = 0$, т.е. $\alpha = \varepsilon$, то е ясно от дефиницията на $\boldsymbol{\Delta}^\star$ и $\boldsymbol{\delta}^\star$, че за всяко $R \subseteq \boldsymbol{Q}$ е изпълнено:
$$\boldsymbol{\Delta}^\star(R, \varepsilon) = R = \boldsymbol{\delta}^\star(R, \varepsilon).$$
- Да приемем, че (3.10) е изпълнено за думи α с дължина n, т.е.
$$(\forall \alpha \in \boldsymbol{\Sigma}^n)(\forall R \subseteq \boldsymbol{Q})[\, \boldsymbol{\Delta}^\star(R, \alpha) = \boldsymbol{\delta}^\star(R, \alpha)\,].$$
- Нека сега α има дължина $n+1$, т.е. $\alpha = \beta a$, където $|\beta| = n$ и $a \in \boldsymbol{\Sigma}$.

$$\begin{aligned}
\boldsymbol{\delta}^\star(R, \beta a) &= \boldsymbol{\delta}(\boldsymbol{\delta}^\star(R, \beta a)) && \text{// деф. на } \delta^\star \\
&= \boldsymbol{\delta}(\boldsymbol{\Delta}^\star(R, \beta), a) && \text{// от } \textbf{(И.П.)} \text{ за } \beta \\
&= \boldsymbol{\Delta}^\star(\boldsymbol{\Delta}^\star(R, \beta), a) && \text{// от деф. на } \delta \\
&= \boldsymbol{\Delta}^\star(R, \beta a). && \text{// от } \textit{Твърдение 3.14}
\end{aligned}$$

Така доказахме, че

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

Това означава, че според принципа на математическата индукция имаме Свойство (3.10).

Сега вече е лесно да съобразим, че

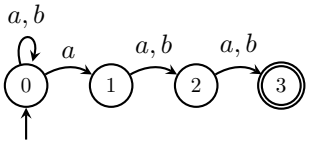
$$\begin{aligned}
\omega \in \mathcal{L}(\mathcal{D}) &\Leftrightarrow \boldsymbol{\delta}^{\star}(\boldsymbol{s},\omega) \in \boldsymbol{F}' && \text{// деф. на } \mathcal{L}(\mathcal{D}) \\
&\Leftrightarrow \boldsymbol{\Delta}^{\star}(\boldsymbol{S},\omega) \cap \boldsymbol{F} \neq \emptyset && \text{// от (3.10)} \\
&\Leftrightarrow \omega \in \mathcal{L}(\mathcal{N}) && \text{// деф. на } \mathcal{L}(\mathcal{N}).
\end{aligned}$$

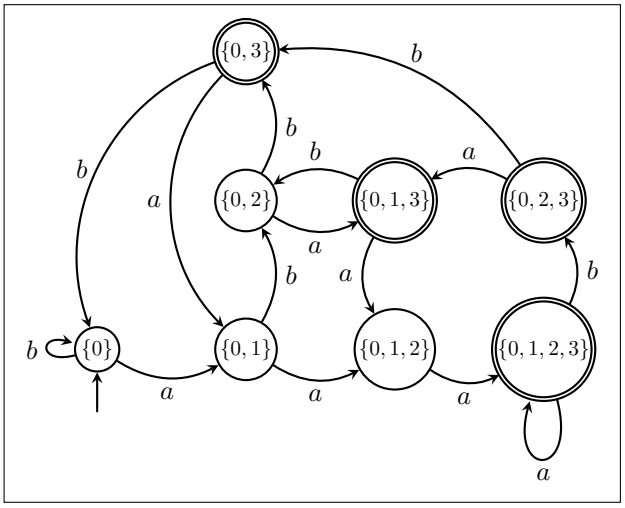
□

Експоненциална експлозия

Да разгледаме недетерминирания краен автомат $\mathcal{N}$ зададен на *Фигура 3.18*. Нека да приложим конструкцията на Рабин-Скот за получаване на ДКА $\mathcal{D}$ разпознаващ езика на $\mathcal{N}$.

Фигура 3.18: Недетерминиран автомат $\mathcal{N}$ за езика съставен думите, в които има a на трета позиция от дясно на ляво.





Фигура 3.19: Детерминиран автомат $\mathcal{D}$ за езика съставен думите, в които има a на трета позиция от дясно на ляво.

Сега ще видим, че не можем да получим детерминиран автомат с по-малко състояния за този език.

Твърдение 3.15. Съществува НКА $\mathcal{N}$ с $n+1$ състояния, за който не съществува ДКА $\mathcal{A}$ с по-малко от 2^n състояния.

Тук следваме [11, стр. 66] и [1, стр. 164]. В [21, стр. 80] има друг пример за НКА с n състояния вместо $n+1$, но доказателството изглежда по-сложно. Езикът L може да се представи и така:

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

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

Упътване. Лесно се съобразява, че за $n > 0$, съществува недетерминиран краен автомат $\mathcal{N}$ с $n+1$ състояния, който разпознава езика

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

На *Фигура 3.18* е изобразен частния случай за $n = 3$.

Нека да съобразим, че не е възможно да съществува ДКА $\mathcal{A}$ разпознаващ същия език L с по-малко от 2^n състояния. Да допуснем, че $|\boldsymbol{Q}_\mathcal{A}| < 2^n$. От принципа на Дирихле имаме, че съществуват две различни думи α и β с дължина n, за които съществува $q \in \boldsymbol{Q}_\mathcal{A}$ и $\boldsymbol{\delta}^\star_\mathcal{A}(\boldsymbol{s}, \alpha) = q = \boldsymbol{\delta}^\star_\mathcal{A}(\boldsymbol{s}, \beta)$. Нека *първата* разлика в тези две думи е на позиция $i < n$, т.е. думите α и β могат да се представят като

$$\alpha = \lambda \cdot \alpha[i] \cdot \rho,$$
$$\beta = \lambda \cdot \beta[i] \cdot \rho'.$$

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

- Нека $i = 0$. Тогава $\rho = \rho' = \varepsilon$. Това означава, че $\alpha \in L$, но $\beta \notin L$. Следователно състоянието q е едновременно финално и нефинално. Това е противоречие.
- Нека $i > 0$. 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 $\boldsymbol{\delta}^\star_{\mathcal{A}}(s, \alpha_0) = p = \boldsymbol{\delta}^\star_{\mathcal{A}}(s, \beta_0)$ и състоянието p трябва да е едновременно финално и нефинално.

□

3.9 Фундаментална теорема

Сега ще видим една естествена характеризация на езиците $\mathcal{L}_{\mathcal{A}}(q)$ като решение на система от уравнения. Този подход ще ни помогне в Раздел 3.10 да направим връзка между автоматните и регулярните езици.

[20, стр. 133], [14, стр. 63].

Лема 3.3. Да разгледаме произволен краен автомат $\mathcal{A}$, за който $\boldsymbol{Q}_{\mathcal{A}} = \{q_1, q_2, \ldots, q_n\}$. За произволни индекси $i, j = 1, 2, \ldots, n$, да положим:

$$R_{i,j} \stackrel{\text{деф}}{=} \{a \in \boldsymbol{\Sigma} \,:\, q_j \in \boldsymbol{\Delta}_{\mathcal{A}}(q_i, a)\}$$

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

Тогава езиците $L_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ задават *единственото* решение на системата:

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

Упътване. Можем да разгледаме всеки ред поотделно. Достатъчно е да докажем, че за произволно i, където $1 \leq i \leq n$, за произволна дума ω е изпълнена еквивалентността:

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

Да фиксираме произволен индекс i. Ще разгледаме два случая за думата ω. Нека първо $|\omega| = 0$, т.е. $\omega = \varepsilon$. Тогава:

$$\begin{aligned} \varepsilon \in \mathcal{L}_{\mathcal{A}}(q_i) \;&\Leftrightarrow\; q_i \in \boldsymbol{F}_{\mathcal{A}} \\ &\Leftrightarrow\; \varepsilon \in P_i && \text{// деф. на } P_i \\ &\Leftrightarrow\; \varepsilon \in \bigcup_{j=1}^{n} R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j) \cup P_i && \text{// } \varepsilon \notin \bigcup_{j=1}^{n} R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j) \end{aligned}$$

Нека сега $|\omega| > 0$, т.е. $\omega = a\beta$. Тогава:

$$\begin{aligned} a\beta \in \mathcal{L}_{\mathcal{A}}(q_i) &\Leftrightarrow \Delta^{\star}_{\mathcal{A}}(\Delta_{\mathcal{A}}(q_i, a), \beta)) \in \boldsymbol{F}_{\mathcal{A}} \\ &\Leftrightarrow (\exists q_j \in \boldsymbol{Q}_{\mathcal{A}})[q_j \in \Delta_{\mathcal{A}}(q_i, a) \;\&\; \Delta^{\star}_{\mathcal{A}}(\{q_j\}, \beta) \in \boldsymbol{F}_{\mathcal{A}}] \\ &\Leftrightarrow (\exists j)[a \in R_{i,j} \;\&\; \beta \in \mathcal{L}_{\mathcal{A}}(q_j)] \\ &\Leftrightarrow a\beta \in \bigcup_{j=1}^{n} R_{i,j} \cdot \mathcal{L}_{\mathcal{A}}(q_j) \cup P_i. \qquad \text{// } a\beta \notin P_i \end{aligned}$$

□

Сега ще видим, че имаме и обратната посока, което означава, че по система можем да построим автомат.

Лема 3.4. Да разгледаме следната произволна система от вида

$$\left|\begin{array}{rcl} X_1 &=& \bigcup\limits_{i=1}^{n} R_{1,i} \cdot X_i \cup P_1 \\ &\vdots& \\ X_n &=& \bigcup\limits_{i=1}^{n} R_{n,i} \cdot X_i \cup P_n \end{array}\right. \tag{3.11}$$

при която $P_i \subseteq \{\varepsilon\}$ и $R_{i,j} \subseteq \boldsymbol{\Sigma}$, за всяко $i, j = 1, \ldots, n$. Да дефинираме НКА $\mathcal{A}$ по следния начин:

- $\boldsymbol{Q}_{\mathcal{A}} \stackrel{\text{деф}}{=} \{q_1, \ldots, q_n\}$,
- $\boldsymbol{F}_{\mathcal{A}} \stackrel{\text{деф}}{=} \{q_i \in \boldsymbol{Q}_{\mathcal{A}} \,:\, \varepsilon \in P_i\}$,
- $\boldsymbol{\Delta}_{\mathcal{A}}(q_i, a) \stackrel{\text{деф}}{=} \{q_j \in \boldsymbol{Q}_{\mathcal{A}} \,:\, a \in R_{i,j}\}$.

Тогава $\mathcal{L}_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ е *единственото* решение на системата (3.11).

Доказателство. Нека имаме автомата $\mathcal{A}$ от условието на лемата. За него по *Лема 3.3* имаме, че езиците $\mathcal{L}_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ задават *единственото* решение на системата

$$\left|\begin{array}{rcl} X_1 &=& \bigcup\limits_{i=1}^{n} \hat{R}_{1,i} \cdot X_i \cup \hat{P}_1 \\ &\vdots& \\ X_n &=& \bigcup\limits_{i=1}^{n} \hat{R}_{n,i} \cdot X_i \cup \hat{P}_n. \end{array}\right. \tag{3.12}$$

Да напомним, че $\hat{R}_{i,j} \stackrel{\text{деф}}{=} \{a \in \Sigma \,:\, q_j \in \boldsymbol{\Delta}_{\mathcal{A}}(q_i, a)\}$, $\hat{P}_i \stackrel{\text{деф}}{=} \{\varepsilon\}$, ако $q_i \in \boldsymbol{F}_{\mathcal{A}}$ и $\hat{P}_i \stackrel{\text{деф}}{=} \emptyset$ в противен случай. Понеже $\varepsilon \in \hat{P}_i \;\Leftrightarrow\; q_i \in \boldsymbol{F}_{\mathcal{A}} \;\Leftrightarrow\; \varepsilon \in P_i$, то е ясно, че

$P_i = \hat{P}_i$. Освен това,

$$a \in \hat{R}_{i,j} \;\Leftrightarrow\; q_j \in \mathbf{\Delta}_{\mathcal{A}}(q_i, a) \;\Leftrightarrow\; a \in R_{i,j}.$$

Заключаваме, че $R_{i,j} = \hat{R}_{i,j}$. Оттук следва, че системата (3.11) съвпада със системата (3.12) и тогава $\mathcal{L}_{\mathcal{A}}(q_1), \ldots, \mathcal{L}_{\mathcal{A}}(q_n)$ е единственото решение на системата (3.11). □

От гледна точка на *Теорема 3.5*, моделът на недетерминираният автомат е естественият модел, а не този на детерминираният.

✍ Съобразете, че един автомат е ДКА точно тогава, когато в неговата система, за всяко i са изпълнени условията:

- тоталност:

$$\bigcup_{j=1}^{n} R_{i,j} = \mathbf{\Sigma}$$

- детерминираност:

$R_{i,j} \cap R_{i,k} = \emptyset$ за всеки $j \neq k$.

Като обединим двете леми, получаваме следната теорема.

Теорема 3.5 (Фундаментална теорема за автоматните езици)**.** Един език L е *автоматен* точно тогава, когато L е измежду езиците $L_1, \ldots, L_n$, които представляват *единственото* решение на система от вида

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

където $P_i \subseteq \{\varepsilon\}$ и $R_{i,j} \subseteq \mathbf{\Sigma}$, за всяко $i, j = 1, \ldots, n$.

3.10 Автоматните езици са регулярни

Следващата стъпка е да видим как системите от уравнения от регулярни езици се „връзват" с автоматните езици. Оказва се, че ние вече знаем как. Да разгледаме произволен ДКА $\mathcal{A}$. Достатъчно е да си припомним *Лема 3.3*. Според нея, езиците $\mathcal{L}_\mathcal{A}(q_1), \ldots, \mathcal{L}_\mathcal{A}(q_n)$ са решение на системата

Тук не е важно дали $\mathcal{A}$ е ДКА или НКА.

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

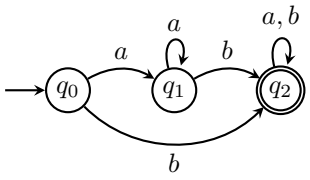
Знаем, че $R_{i,j} \stackrel{\text{деф}}{=} \{a \in \boldsymbol{\Sigma} : \boldsymbol{\delta}(q_i, a) = q_j\}$ е краен език и следователно е регулярен. Ясно е също така, че $\varepsilon \notin R_{i,j}$. Знаем също, че

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

които отново са крайни езици и следователно регулярни. Според *Следствие 2.2* 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, и понеже решението е единствено, 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 се оказа, че ние безплатно получихме следната теорема.

Теорема 3.6 (Клини [12]). Всеки автоматен език е регулярен.

Фигура 3.20: Ще проверим, че $\mathcal{L}(\mathcal{A})$ се описва с регулярния израз $\boldsymbol{a^\star b(a+b)^\star}$.



Пример 3.2. Да разгледаме отново автомата от *Фигура 3.20*. На него съответства следната система от уравнения на езици:

$$\left|\begin{array}{rcl} \mathcal{L}_\mathcal{A}(q_0) &=& \{a\} \cdot \mathcal{L}_\mathcal{A}(q_1) \cup \{b\} \cdot \mathcal{L}_\mathcal{A}(q_2) \cup \emptyset \\ \mathcal{L}_\mathcal{A}(q_1) &=& \{a\} \cdot \mathcal{L}_\mathcal{A}(q_1) \cup \{b\} \cdot \mathcal{L}_\mathcal{A}(q_2) \cup \emptyset \\ \mathcal{L}_\mathcal{A}(q_2) &=& \{a\} \cdot \mathcal{L}_\mathcal{A}(q_2) \cup \{b\} \cdot \mathcal{L}_\mathcal{A}(q_2) \cup \{\varepsilon\}. \end{array}\right. \tag{3.13}$$

За простота, да преобразуваме (3.13) в система от уравнения на регулярни изрази:

$$\left|\begin{array}{rcl} \boldsymbol{r}_0 &=& \boldsymbol{a} \cdot \boldsymbol{r}_1 + \boldsymbol{b} \cdot \boldsymbol{r}_2 + \boldsymbol{\emptyset} \\ \boldsymbol{r}_1 &=& \boldsymbol{a} \cdot \boldsymbol{r}_1 + \boldsymbol{b} \cdot \boldsymbol{r}_2 + \boldsymbol{\emptyset} \\ \boldsymbol{r}_2 &=& \boldsymbol{a} \cdot \boldsymbol{r}_2 + \boldsymbol{b} \cdot \boldsymbol{r}_2 + \boldsymbol{\varepsilon}. \end{array}\right.$$

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

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 в нашия пример $r_0 = r_1$. По-късно ще видим, че това означава, че можем да „слеем" двете състояния и да получим по-компактен автомат за същия език.
За подробно изложение на въпроса как можем да извлечем алгоритъм, вижте [22, стр. 69]. Идеята е на практика изложена в Пример 3.2.

Заключаваме, че езикът на $\mathcal{A}$ 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 3.16. Съществува алгоритъм, за който при вход краен автомат $\mathcal{A}$, извежда като изход регулярен израз r, такъв че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(r)$.

3.11 Регулярните езици са автоматни (*)

Да напомним, че от *Теорема 2.2* знаем, че Q_L е крайно множество, когато L е регулярен език. Тогава пък от *Твърдение 3.6* следва, че каноничният автомат $\mathcal{B}_L$ е ДКА и разпознава точно езикът L. Оттук следва, че ние отдавна знаем, че регулярните езици са автоматни.

Тук ще докажем, че регулярните езици са автоматни с индукция по построението на регулярните езици. Ползата от този подход е, че ще получим общи конструкции на автомати за основните регулярни операции - обединение, конкатенация и звезда на Клини. Да започнем с базата на индукцията.

Лема 3.5. Езиците $\emptyset$, $\{\varepsilon\}$, $\{a\}$ за всяко $a \in \Sigma$ са автоматни.

Упътване. *Фигура 3.21* показва недетерминирани крайни автомати за езиците $\emptyset$, $\{\varepsilon\}$ и $\{a\}$. Те не отговарят на нашата дефиниция за детерминиран краен автомат, защото функциите на преходите им не са тотални. □

Сега преминаваме към индукционната стъпка. Това означава, че при дадени произволни автоматни езици L_1 и L_2, то $L_1 \cup L_2$, $L_1 \cdot L_2$ и $L_1^\star$ са автоматни езици.

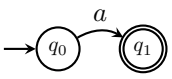
Лема 3.6. Автоматните езици са затворени относно операцията обединение.

Упътване. Това означава, че ако два езика L_1 и L_2 се разпознават от крайни автомати, то трябва да докажем, че $L_1 \cup L_2$ се разпознава от краен автомат. Понеже приемаме, че недетерминираните крайни автомати могат да имат множество от начални състояния, то автомат за $L_1 \cup L_2$ може да се построи като просто обединим състоянията на автоматите за L_1 и L_2 както и функциите на преходите и множествата от начални и финални състояния. Тази проста конструкция е илюстрирана на *Фигура 3.22*. □

Фигура 3.21: Автомати за базовите случаи от дефиницията на регулярните езици:

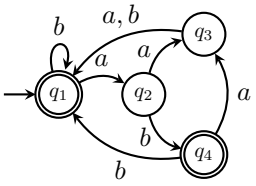
(а) $L(\mathcal{N}) = \emptyset$

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

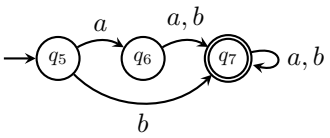


(в) $\mathcal{L}(\mathcal{N}) = \{a\}$

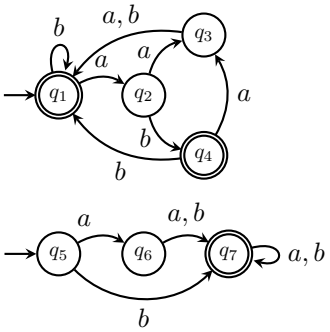
Фигура 3.22: Автомат за обединението получаваме като просто обединим компонентите на два автомата:



(а) Автомат $\mathcal{A}_1$



(б) Автомат $\mathcal{A}_2$



(в) Недетерминиран автомат за $\mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$ с начални състояния $\{q_1, q_5\}$.

Лема 3.7. Автоматните езици са затворени относно операцията конкатенация.

Да започнем с един автомат $\mathcal{A}_1$ с n състояния разпознаващ езика $L_1 = \mathcal{L}_{\mathcal{A}_1}(\boldsymbol{s}_1)$. Знаем, че на него съответства системата

$$\left|\begin{array}{rcl} X_1 & = & \bigcup\limits_{j=1}^{n} R_{1,j} \cdot X_j \ \cup \ P_1 \\ & \vdots & \\ X_n & = & \bigcup\limits_{j=1}^{n} R_{n,j} \cdot X_j \ \cup \ P_n, \end{array}\right. \tag{3.14}$$

чието *единствено* решение е n-орка от езици $L_1, \ldots, L_n$.

За да бъдат езиците $L_1, \ldots, L_n$ решението на системата за автомата $\mathcal{A}_1$, означава следното:

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

Да вземем и друг автомат $\mathcal{A}_2$ с k състояния разпознаващ езика $M_1 = \mathcal{L}_{\mathcal{A}_2}(\boldsymbol{s}_2)$. Знаем, че на него съответства системата

$$\left|\begin{array}{rcl} Y_1 & = & \bigcup\limits_{j=1}^{k} T_{1,j} \cdot Y_j \ \cup \ E_1 \\ & \vdots & \\ Y_k & = & \bigcup\limits_{j=1}^{k} T_{k,j} \cdot Y_j \ \cup \ E_k, \end{array}\right. \tag{3.15}$$

чието *единствено* решение е k-орка от езици $M_1, \ldots, M_k$.

За да бъдат езиците $M_1, \ldots, M_k$ решението на системата за автомата $\mathcal{A}_2$, означава следното:

$$\left|\begin{array}{rcl} M_1 & = & \bigcup\limits_{i=1}^{k} T_{1,i} \cdot M_i \cup E_1 \\ & \vdots & \\ M_k & = & \bigcup\limits_{i=1}^{k} T_{k,i} \cdot M_i \cup E_k. \end{array}\right.$$

Ще дефинираме нова система от $(n+k)$ уравнения, чието единствено решение ще бъдат езиците $L_1M_1, L_2M_1, \ldots, L_nM_1, M_1, \ldots, M_k$. Понеже за $i = 1, \ldots, n$ имаме равенствата

$$L_i = \bigcup_{j=1}^{n} R_{i,j} \cdot L_j \ \cup \ P_i,$$

ясно е, че можем да конкатенираме навсякъде отдясно с M_1 и да получим:

$$L_iM_1 = \bigcup_{j=1}^{n} R_{i,j} \cdot L_jM_1 \ \cup \ P_iM_1,$$

Следната n-орка от езици

$$L_1M_1, \ldots, L_nM_1$$

задава единственото решение на системата:

$$\left|\begin{array}{rcl} X_1 & = & \bigcup\limits_{i=1}^{n} R_{1,i} \cdot X_i \cup P_1M_1 \\ & \vdots & \\ X_n & = & \bigcup\limits_{i=1}^{n} R_{n,i} \cdot X_i \cup P_nM_1. \end{array}\right.$$

Проблемът е, че от тази система не можем да получим автомат. Трябва да е преобразуваме във вид, при който можем да се позовем на Фундаменталната теорема за автоматни езици.

За произволно $i = 1, \ldots, n$ имаме два случая, които трябва да разгледаме.

- Ако $P_i = \emptyset$, то е ясно, че $P_iM_1 = P_i$. Тогава имаме, че

$$L_iM_1 = \bigcup_{j=1}^{n} R_{i,j} \cdot L_jM_1 \ \cup \ P_i.$$

- Ако $P_i = \{\varepsilon\}$, то е ясно, че $P_iM_1 = M_1$. Тогава използваме дясната страна на

равенството за M_1 за да получим равенството:

$$L_iM_1 = \bigcup_{j=1}^{n} R_{i,j} \cdot L_jM_1 \ \cup \ \underbrace{\bigcup_{j=1}^{k} T_{1,j} \cdot M_j \ \cup \ E_1}_{M_1}.$$

От тези две наблюдения следва, че можем да представим уравненията по следния начин:

$$L_iM_1 = \bigcup_{j=1}^{n} R_{i,j} \cdot L_jM_1 \ \cup \ \bigcup_{j=1}^{k} \hat{T}_{i,j}M_j \ \cup \ \hat{E}_i,$$

където за $i = 1, \ldots, n$ и $j = 1, \ldots, k$ сме положили:

Или с други думи,

$\hat{T}_{i,j} = T_{1,j} \cdot P_i,$
$\hat{E}_i = E_1 \cdot P_i.$

$$\hat{T}_{i,j} \stackrel{\text{деф}}{=} \begin{cases} T_{1,j}, & \text{ако } P_i = \{\varepsilon\} \\ \emptyset, & \text{ако } P_i = \emptyset, \end{cases}$$

$$\hat{E}_i \stackrel{\text{деф}}{=} \begin{cases} E_1, & \text{ако } P_i = \{\varepsilon\} \\ \emptyset, & \text{ако } P_i = \emptyset. \end{cases}$$

Сега можем да обединим всичко направено дотук в една система с $n + k$ уравнения и $n + k$ неизвестни:

$$\left| \begin{array}{rcl} X_i & = & \bigcup\limits_{j=1}^{n} R_{i,j} \cdot X_j \ \cup \ \bigcup\limits_{j=1}^{k} \hat{T}_{i,j}Y_j \ \cup \ \hat{E}_i \\ Y_\ell & = & \bigcup\limits_{j=1}^{n} \emptyset \cdot X_j \ \cup \ \bigcup\limits_{j=1}^{k} T_{\ell,j} \cdot Y_j \ \cup \ E_\ell, \end{array} \right.$$

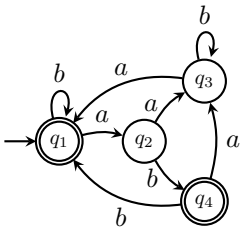
където $i = 1, \ldots, n$ и $\ell = 1, \ldots, k$.

Тази система има *единствено* решение съставено от $(n + k)$-орката от езици

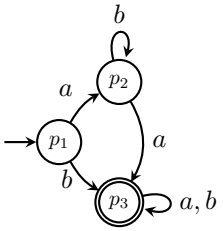
$$L_1M_1, \ldots, L_nM_1, M_1, \ldots, M_k.$$

Директно от тази конструкция получаваме алгоритъм за намиране на автомат за конкатенацията на двата дадени първоначално автомата. Ползата от този подход е, че понеже на всяка стъпка правим еквивалентни преобразования, то безплатно получаваме коректността на конструкцията на автомат за конкатенацията.

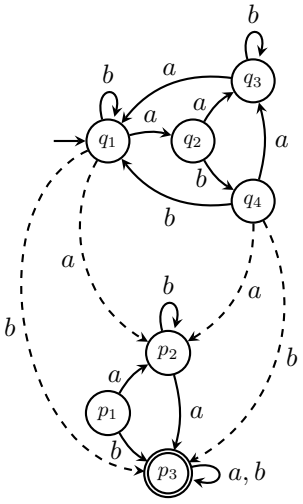
Фигура 3.23: Конкатенация на автомати според доказателството на *Лема 3.7*



(а) автомат $\mathcal{A}_1$ за езика L_1.



(б) автомат $\mathcal{A}_2$ за езика L_2.



(в) Автомат $\mathcal{A}$ за езика $L_1 \cdot L_2$.

Пример 3.3. Нека да видим как можем да приложим горната конструкция за конкатенация на езиците на два конкретни автомата. Да започнем с автомата $\mathcal{A}_1$ изобразен на *Фигура 3.23а*. На него съответства следната система:

$$\left|\begin{array}{lcl} X_1 &=& \{a\}X_2 \cup \{b\}X_1 \cup \{\varepsilon\} \\ X_2 &=& \{a\}X_3 \cup \{b\}X_4 \\ X_3 &=& \{a\}X_1 \cup \{b\}X_3 \\ X_4 &=& \{a\}X_3 \cup \{b\}X_1 \cup \{\varepsilon\}. \end{array}\right.$$

Да разгледаме също така и автомата $\mathcal{A}_2$ изобразен на *Фигура 3.23б*. На него съответства следната система:

$$\left|\begin{array}{lcl} Y_1 &=& \{a\}Y_2 \cup \{b\}Y_3 \\ Y_2 &=& \{a\}Y_3 \cup \{b\}Y_2 \\ Y_3 &=& \{a\}Y_3 \cup \{b\}Y_3 \cup \{\varepsilon\}. \end{array}\right.$$

От автоматна гледна точка, трябва да свържем финалните състояния на $\mathcal{A}_1$ с *наследниците* на началното състояние на $\mathcal{A}_2$, като внимаваме да запазим етикетите по новите преходи. От алгебрична гледна точка, просто заместваме всяко срещане на $\{\varepsilon\}$ в системата за $\mathcal{A}_1$ с дефиницията на Y_1. Така получаваме следната система:

$$\left|\begin{array}{lcl} X_1 &=& \{a\}X_2 \cup \{b\}X_1 \cup \boldsymbol{\{a\}Y_2 \cup \{b\}Y_3} \\ X_2 &=& \{a\}X_3 \cup \{b\}X_4 \\ X_3 &=& \{a\}X_1 \cup \{b\}X_3 \\ X_4 &=& \{a\}X_3 \cup \{b\}X_1 \cup \boldsymbol{\{a\}Y_2 \cup \{b\}Y_3} \\ Y_1 &=& \{a\}Y_2 \cup \{b\}Y_3 \\ Y_2 &=& \{a\}Y_3 \cup \{b\}Y_2 \\ Y_3 &=& \{a\}Y_3 \cup \{b\}Y_3 \cup \{\varepsilon\}. \end{array}\right.$$

За нея директно получаваме автомата $\mathcal{A}$ на *Фигура 3.23в*.

Лема 3.8. Автоматните езици са затворени относно звезда на Клини.

Доказателство. Ще покажем, че автоматините езици са затворени относно операцията плюс. След това лесно се вижда защо автоматните езици са затворени и относно операцията звезда. Да започнем от един автомат $\mathcal{A}$ разпознаващ езика $L_1 = \mathcal{L}_{\mathcal{A}}(s)$. Знаем, че на него съответства системата

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

чието *единствено* решение е n-орка от езици $L_1, \ldots, L_n$.

Да напомним, че $R_{i,j} \subseteq \mathbf{\Sigma}$ и $P_j \subseteq \{\varepsilon\}$. Освен това, щом езиците $L_1, \ldots, L_n$ задават решението на системата, то имаме:

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

Ще видим, че като направим редица еквивалентни преобразувания върху тази система от уравнения, ще получим нова система, чието *единствено* решение ще бъде n-орката от езици $L_1L_1^\star, L_2L_1^\star, \cdots, L_nL_1^\star$. Тогава от фундаменталната теорема за автомати ще следва, че имаме автомат $\hat{\mathcal{A}}$ разпознаващ езика L_1^+. Първата стъпка е да конкатенираме навсякъде отдясно с $L_1^\star$ за да получим следните равенства за $i = 1, \ldots, n$:

$$L_iL_1^\star = \bigcup_{j=1}^{n} R_{i,j} \cdot L_jL_1^\star \ \cup \ P_i \cdot L_1^\star.$$

Ясно е, че членовете $P_i \cdot L_1^\star$ заслужават внимание. За да преведем новата система от уравнения в подходящия за нас вид, трябва да разгледаме два случая в зависимост от това дали $P_i = \emptyset$ или $P_i = \{\varepsilon\}$.

Нека първо да разгледаме случая, когато $P_i = \emptyset$, за някое i. Тогава $P_i \cdot L_1^\star = P_i$. Получаваме, че

$$L_iL_1^\star = \bigcup_{j=1}^{n} R_{i,j} \cdot L_jL_1^\star \ \cup \ \underbrace{P_i}_{\emptyset}.$$

Сега да разгледаме случая, когато $P_i = \{\varepsilon\}$. Тогава е ясно, че $P_iL_1 = L_1$. Първо да направим наблюдението, че за произволен език L е изпълнено равенството

$$L^\star = (L \setminus \{\varepsilon\}) \cdot L^\star \cup \{\varepsilon\}. \tag{3.16}$$

По този начин „разгръщаме” рекурсията. Обръщането към L_1 се замества с неговата дефиниция.

Оттук получаваме следната верига от равенства:

$$\underbrace{P_iL_1^\star}_{L_1^\star} = (L_1 \setminus \{\varepsilon\})L_1^\star \cup \underbrace{\{\varepsilon\}}_{P_i} \qquad \text{// от 3.16}$$

$$= (\underbrace{\bigcup_{j=1}^{n} R_{1,j} \cdot L_j}_{L_1 \setminus \{\varepsilon\}}) \cdot L_1^\star \cup P_i$$

$$= \bigcup_{j=1}^{n} R_{1,j} \cdot L_j L_1^\star \ \cup \ P_i.$$

✍ Тук единственото, което правим е да приведем системата във вид, за който можем да приложим фундаменталната теорема за автоматни езици.
✍ Защо това доказателство е достатъчно? В случай, че $\varepsilon \in L_1$, то ние сме готови, защото тогава $L_1^+ = L_1^\star$. В противен случай, добавяме новото уравнение $X_0 = \{\varepsilon\}$ и обявяваме, че началните състояния на новополучения автомат, разпознаващ $L_1^\star$, ще бъдат съответстващите на променливите X_0 и X_1. Така получаваме новата система:

Накрая заключаваме, че i-тия ред на системата може да се запише така:

$$L_iL_1^\star = \bigcup_{j=1}^{n} R_{i,j} \cdot L_j L_1^\star \ \cup \ P_iL_1^\star$$

$$= \bigcup_{j=1}^{n} R_{i,j} \cdot L_j L_1^\star \ \cup \ \underbrace{\bigcup_{j=1}^{n} R_{1,j} \cdot L_j L_1^\star \ \cup \ P_i}_{P_iL_1^\star}$$

$$= \bigcup_{j=1}^{n} \underbrace{(R_{i,j} \cup R_{1,j})}_{\hat{R}_{i,j}} \cdot L_j L_1^\star \ \cup \ P_i.$$

$$\left|\begin{array}{lcl} X_0 & = & \bigcup\limits_{j=0}^{n} \hat{R}_{0,j} \cdot X_j \cup P_0 \\ X_1 & = & \bigcup\limits_{j=0}^{n} \hat{R}_{1,j} \cdot X_j \ \cup \ P_1 \\ & \vdots & \\ X_n & = & \bigcup\limits_{j=0}^{n} \hat{R}_{n,j} \cdot X_j \ \cup \ P_n, \end{array}\right.$$

където $P_0 \stackrel{\text{деф}}{=} \{\varepsilon\}$ и $\hat{R}_{i,j} \stackrel{\text{деф}}{=} \emptyset$, ако $i = 0$ или $j = 0$.

Нека положим

$$\hat{R}_{i,j} \stackrel{\text{деф}}{=} \begin{cases} R_{i,j}, & \text{ако } P_i = \emptyset \\ R_{i,j} \cup R_{1,j}, & \text{ако } P_i = \{\varepsilon\}, \end{cases}$$

или с други думи,

$$\hat{R}_{i,j} = R_{i,j} \cup P_i \cdot R_{1,j}.$$

Заключаваме, че *единственото* решение на новата система

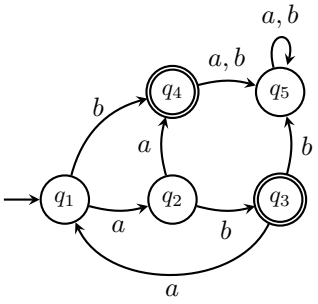
$$\left|\begin{array}{lcl} X_1 & = & \bigcup\limits_{j=1}^{n} \hat{R}_{1,j} \cdot X_j \ \cup \ P_1 \\ & \vdots & \\ X_n & = & \bigcup\limits_{j=1}^{n} \hat{R}_{n,j} \cdot X_j \ \cup \ P_n \end{array}\right.$$

е n-орката от езици $L_1L_1^\star, L_2L_1^\star, \ldots, L_nL_1^\star$. □

От доказателството на *Лема 3.8* директно получаваме алгоритъм за намиране на автомат разпознаващ звездата на Клини на езика на входния автомат. Ползата от нашият подход е, че няма нужда да доказваме коректност на този алгоритъм.

Пример 3.4. Нека да видим как можем да приложим горната конструкция за звезда на език върху конкретен автомат. Да започнем с автомата $\mathcal{A}_1$ изобразен на *Фигура 3.24а*. Според фундаменталната теорема, за него имаме следната система:

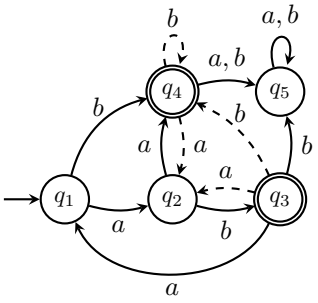
$$\left|\begin{array}{lcl} X_1 & = & \{a\}X_2 \cup \{b\}X_4 \\ X_2 & = & \{a\}X_4 \cup \{b\}X_3 \\ X_3 & = & \{a\}X_1 \cup \{b\}X_5 \cup \{\varepsilon\} \\ X_4 & = & \{a,b\}X_5 \cup \{\varepsilon\} \\ X_5 & = & \{a,b\}X_5 \end{array}\right.$$



(а) автомат $\mathcal{A}$ за езика L.

Да видим как можем да намерим автомат за L^+. От автоматна гледна точка, свързваме финалните състояния на $\mathcal{A}$ с наследниците на неговото начално състояние, като внимаваме да запазим етикетите по новите преходи. От алгебрична гледна точка, просто копираме дефиницията на X_1 на тези редове, където имаме $\{\varepsilon\}$. Така получаваме системата:

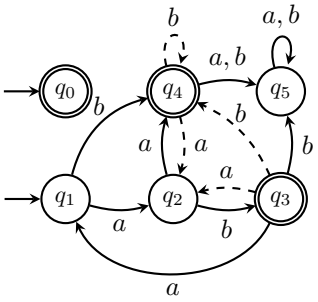
$$\left|\begin{array}{lcl} X_1 & = & \{a\}X_2 \cup \{b\}X_4 \\ X_2 & = & \{a\}X_4 \cup \{b\}X_3 \\ X_3 & = & \{a\}X_1 \cup \{b\}X_4 \cup \boldsymbol{\{a\}X_2 \cup \{b\}X_4} \cup \{\varepsilon\} \\ X_4 & = & \{a,b\}X_5 \cup \boldsymbol{\{a\}X_2 \cup \{b\}X_4} \cup \{\varepsilon\} \\ X_5 & = & \{a,b\}X_5 \end{array}\right.$$



(б) Автомат $\mathcal{A}^+$ за езика L^+.

На тази система отговаря автомата $\mathcal{A}^+$ изобразен на *Фигура 3.24б*. Оттук вече е ясно как можем да получим автомат за $L^\star$ в случай, че $\varepsilon \notin L$. Просто добавяме ново начално състояние, което едновременно с това е и финално и от което не излизат преходи. Ако q_1 беше финално, то нямаше да има нужда да добавяме q_0. Това е окончателната система:

$$\left|\begin{array}{lcl} X_0 & = & \{\varepsilon\} \\ X_1 & = & \{a\}X_2 \cup \{b\}X_4 \\ X_2 & = & \{a\}X_4 \cup \{b\}X_3 \\ X_3 & = & \{a\}X_1 \cup \{b\}X_4 \cup \{a\}X_2 \cup \{b\}X_4 \cup \{\varepsilon\} \\ X_4 & = & \{a,b\}X_5 \cup \{a\}X_2 \cup \{b\}X_4 \cup \{\varepsilon\} \\ X_5 & = & \{a,b\}X_5 \end{array}\right.$$



(в) Автомат $\mathcal{A}^\star$ за езика $L^\star$.

На нея съответства недетерминиран краен автомат $\mathcal{A}^\star$ с начални състояния $\{q_0, q_1\}$. Този автомат е изобразен на *Фигура 3.24в*.

3.12 Трети критерий за регулярност (*)

Лема 3.9 (Лема за покачването). Нека L да бъде *безкраен* автоматен език. Съществува число $p \geq 1$, зависещо само от L, за което за всяка дума $\omega \in L$, за която $|\omega| \geq p$, може да бъде записана като $\omega = xyz$ и

1) $|y| \geq 1$;

2) $|xy| \leq p$;

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

В този раздел ще изложим автоматния поглед върху въпроса за регулярност или „автоматност" на даден език.
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 $xy^0 z = xz$.
Тази лема я има на практика във всеки добър учебник по този предмет. Например, [15, стр. 88], [22, 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

$$\mathcal{A} = \langle \boldsymbol{\Sigma}, \boldsymbol{Q}, \boldsymbol{s}, \boldsymbol{\delta}, \boldsymbol{F} \rangle$$

е краен детерминиран автомат, за който $L = \mathcal{L}(\mathcal{A})$. Да положим $p = |\boldsymbol{Q}|$ и нека ω е дума, за която $|\omega| = k \geq p$. Да разгледаме изпълнението на ω върху $\mathcal{A}$:

$$\boldsymbol{s} \overset{\omega[:p]}{\rightsquigarrow} q_p \overset{\omega[p:]}{\rightsquigarrow} q_k$$

и по специално първите p стъпки. Понеже в това доказателство ще работим с индексите на състоянията, нека за улеснение да положим $q_0 = \boldsymbol{s}$. Тъй като $|\boldsymbol{Q}| = p$, а по пътя

$$q_0 \overset{a_0}{\to} q_1 \overset{a_1}{\to} q_2 \cdots \overset{a_{p-1}}{\to} q_p$$

участват $p+1$ състояния $q_0, q_1, \ldots, q_p$, то съществуват индекси i, j, за които $0 \leq i < j \leq p$ и $q_i = q_j$. Нека разделим думата ω на три части по следния начин:

$$q_0 \overset{\omega[:i]}{\rightsquigarrow} q_i \overset{\omega[i:j]}{\rightsquigarrow} q_j \overset{\omega[j:p]}{\rightsquigarrow} q_p$$

Нека положим $x \overset{\text{деф}}{=} \omega[:i]$, $y \overset{\text{деф}}{=} \omega[i:j]$, $z \overset{\text{деф}}{=} \omega[j:]$. 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 $xy^0 z = xz \in L$, защото имаме следното изчисление:

$$q_0 \overset{\omega[:i]}{\rightsquigarrow} \underbrace{q_i}_{q_j} \overset{\omega[j:]}{\rightsquigarrow} q_k \in F.$$

защото $q_i = q_j$, или с други думи,

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

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

$$q_0 \overset{\omega[:i]}{\rightsquigarrow} q_i \overset{\omega[i:j]}{\rightsquigarrow} \underbrace{q_i}_{q_j} \overset{\omega[i:j]}{\rightsquigarrow} q_j \overset{\omega[j:]}{\rightsquigarrow} q_k \in \boldsymbol{F},$$

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:

$$q_0 \overset{\omega[:i]}{\rightsquigarrow} \underbrace{q_i \overset{\omega[i:j]}{\rightsquigarrow} q_i \overset{\omega[i:j]}{\rightsquigarrow} q_i \cdots \overset{\omega[i:j]}{\rightsquigarrow} \cdots q_i}_{n \text{ пъти}} \overset{\omega[j:]}{\rightsquigarrow} q_k \in \boldsymbol{F},$$

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 $xy^nz \in L$. □

Упражнение по логика

Подобно е изложението и в [13, стр. 70]. Използваме, че съждителната формула $p \to q$ е еквивалентна на $\neg p \vee q$. Думите „разширение" и „покачване" не описват точно свойството на думата α, защото при $i = 0$ на практика думата α се съкращава. Затова е добре винаги да имаме едно наум за случая $i = 0$.

Нека да разгледаме формулата $\texttt{Pump}_L(p, \alpha)$ която ни казва, че думата α може да се „разшири" в езика L с ограничението зададено от p.

$$\texttt{Pump}_L(p, \alpha) \stackrel{\text{деф}}{=} (\exists x)(\exists y)(\exists z)[\alpha = xyz \ \wedge \ |xy| \leq p \ \wedge \ |y| \geq 1 \ \wedge \ (\forall i)[xy^i z \in L]],$$

Да видим какво означава една дума α да не може да се „разшири".

$$\begin{aligned}\neg\texttt{Pump}_L(p, \alpha) &= (\forall x)(\exists y)(\exists z)[\neg(\alpha = xyz \ \wedge \ |xy| \leq p \ \wedge \ |y| \geq 1) \vee \neg(\forall i)[xy^i z \in L]] \\ &= (\forall x)(\exists y)(\exists z)[(\alpha = xyz \ \wedge \ |xy| \leq p \ \wedge \ |y| \geq 1) \to (\exists i)[xy^i z \notin L]].\end{aligned}$$

Сега да разгледаме следната затворена формула

$$\texttt{Pump}_L \stackrel{\text{деф}}{=} (\exists p)[p \geq 1 \ \wedge (\forall \alpha)[(\alpha \in L \ \wedge \ |\alpha| \geq p) \to \texttt{Pump}_L(p, \alpha)]],$$

която на практика дословно превежда свойството от лемата за покачването на езика на логиката. Тогава:

$$\begin{aligned}\neg\texttt{Pump}_L &= \neg(\exists p)[p \geq 1 \ \wedge (\forall \alpha)[\neg(\alpha \in L \ \wedge \ |\alpha| \geq p) \vee \texttt{Pump}_L(p, \alpha)]] \\ &= (\forall p)[\neg p \geq 1 \vee (\exists \alpha)[\neg\neg(\alpha \in L \ \wedge \ |\alpha| \geq p) \wedge \neg\texttt{Pump}_L(p, \alpha)]] \\ &= (\forall p)[p \geq 1 \to (\exists \alpha)[\alpha \in L \ \wedge \ |\alpha| \geq p \wedge \neg\texttt{Pump}_L(p, \alpha)]].\end{aligned}$$

Нека L е произволен безкраен език. Понеже лемата за покачването гласи *„Ако L е регулярен език, то $\texttt{Pump}_L$ е изпълнено"*, в контрапозиция тя има следния вид: *„Ако $\neg\texttt{Pump}_L$ е изпълнено, то L не е регулярен език"*. Това означава, че можем да формулираме лемата за покачването ето така.

Формулата $p \to q$ е еквивалентна на формулата $\neg q \to \neg p$.

Следствие 3.4. Нека L е произволен безкраен език. Нека също така:

(∀) 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 $xy^i z \notin L$.

Тогава L **не** е регулярен език.

Пример 3.5. Да видим защо езикът $L = \{a^n b^n \ : \ n \in \mathbb{N}\}$ не е регулярен. За да направим това като използваме лемата за покачването, то трябва да докажем, че $\neg\texttt{Pump}_L$ 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 лемата за покачването 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*:

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

Примерни задачи

Пример 3.6. $L \stackrel{\text{деф}}{=} \{a^m b^n \ : \ m < n\}$ не е регулярен.

Доказателство. Доказателството следва стъпките:

(∀) 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 $|y| \geq 1$.

(∃) Ще намерим конкретно $i \in \mathbb{N}$, за което $xy^iz \notin L$. Тук изборът на i не зависи от изборите, които сме направили на предишните стъпки. 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 лемата за покачването следва, че L не е регулярен език. □

Забележка. 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, който все пак зависи от изборите, които сме направили на предишни стъпки. В последния пример, изборът на i не зависеше от предишните избори, които сме направили.

Пример 3.7. $L \stackrel{\text{деф}}{=} \{a^n \ : \ n \text{ е просто}\}$ 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. На последната стъпка ще видим защо този избор е важен за нашите разсъждения.

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

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

е съставно число, следователно $xy^iz \notin L$. Обърнете внимание, че тук изборът на i зависи от предишната стъпка, на която сме разбили думата ω на три части.

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

Пример 3.8. $L \stackrel{\text{деф}}{=} \{a^{n^2} \ : \ n \in \mathbb{N}\}$ не е регулярен.

Доказателство. Тук ще използваме свойството:

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

Доказателството следва стъпките:

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

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

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

(∃) 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 лемата за покачването следва, че L не е регулярен език. □

Пример 3.9. $L \stackrel{\text{деф}}{=} \{a^{n!} \ : \ n \in \mathbb{N}\}$ не е регулярен.

Доказателство. 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 $\omega = a^{(p+1)!}$.

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

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

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

Да разгледаме $i = 2$. Тогава:

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

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

Забележка. Възможно е да вземем $\omega = a^{(p+2)!}$. Да видим възможно ли е $xy^0z \in L$. 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 $|y| \leq p$.

Пример 3.10. $L \stackrel{\text{деф}}{=} \{\alpha\beta \in \{a,b\}^\star \,:\, |\alpha| = |\beta| \;\&\; \alpha \neq \beta\}$ 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

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

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

$$L_2 = L_1 \cap \mathcal{L}(\boldsymbol{a^\star ba^\star b}) = \{a^nba^nb \,:\, n \in \mathbb{N}\}.$$

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

Пример 3.11. $L \stackrel{\text{деф}}{=} \{a^nb^k \,:\, n \neq k\}$ не е регулярен.

Доказателство. 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.

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

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

□

Забележка. Последната задача е добър пример защо е добре първо да се опитаме да преобразуваме дадения език с операции, които запазват регулярността. Тогава можем далеч по-лесно да решим тази задача. Знаем, че регулярните езици са затворени относно допълнение. Следователно, ако допуснем, че L е регулярен, то езикът $L' \stackrel{\text{деф}}{=} \{a\}^\star \cdot \{b\}^\star \setminus L$ също трябва да е регулярен. Но $L' = \{a^nb^n \,:\, n \in \mathbb{N}\}$, 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 $\texttt{Pump}_L$".

Естествено е да се запитаме дали имаме обратната посока на горната импликация, т.е. вярно ли е, че:

„Ако $\texttt{Pump}_L$ е изпълнено, то L е регулярен"?

Сега ще видим, че можем да посочим език L, за който условието $\texttt{Pump}_L$ е изпълнено, но въпреки това L не е регулярен. 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 лемата за покачването. По-късно ще видим един пълен критерий за проверка за регулярност на език.

Пример 3.12. $L \overset{\text{деф}}{=} \{c\}^+ \cdot \{a^n b^n \ : \ n \in \mathbb{N}\} \cup \{a\}^\star \cdot \{b\}^\star$ *не* е регулярен, но условието $\texttt{Pump}_L$ е изпълнено.

За да покажем, че $\texttt{Pump}_L$ 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 $xy^iz \in L$.

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

$$L_1 = L \cap \mathcal{L}(\boldsymbol{c} \cdot \boldsymbol{a}^\star \cdot \boldsymbol{b}^\star) = \{ca^n b^n \ : \ n \in \mathbb{N}\}$$

е регулярен, но с лемата за покачването лесно се вижда, че L_1 не е. Сега да проверим, че $\texttt{Pump}_L$ е изпълнено. Достатъчно е да следваме стъпките:

(∃) 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 $xy^iz \in L$.

□

Приложения

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. [11, стр. 93].

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 3.17. Нека $\mathcal{A}$ е произволен ДКА. Тогава следните условия са еквивалентни:

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

(2) $\mathcal{A}$ разпознава дума ω, за която $|\omega| < |\boldsymbol{Q}|$.

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

$(1) \Rightarrow (2)$. Нека L е непразен език и нека $m = \min\{|\omega| \,:\, \omega \in L\}$. Ще докажем, че $m < |\boldsymbol{Q}|$. За целта, да допуснем, че $m \geq |\boldsymbol{Q}|$ и да изберем $\omega \in L$, за която $|\omega| = m$. Според доказателството на лемата за покачването, съществува разбиване $xyz = \omega$, такова че $xz \in L$. При положение, че $|y| \geq 1$, то $|xz| < m$, което е противоречие с минималността на думата ω. Заключаваме, че нашето допускане е грешно. Тогава $m < |\boldsymbol{Q}|$, откъдето следва, че съществува дума $\omega \in L$ с $|\omega| < |\boldsymbol{Q}|$.

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

□

Твърдение 3.18. Нека $\mathcal{A}$ е произволен ДКА. Тогава следните условия са еквивалентни:

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

(2) $\mathcal{A}$ разпознава дума ω, за която $|\boldsymbol{Q}| \leq |\omega| < 2|\boldsymbol{Q}|$.

Доказателство. Да положим $L = \mathcal{L}(\mathcal{A})$.

$(1) \Rightarrow (2)$**.** Нека L е безкраен език и да вземем *най-късата* дума $\omega \in L$, за която $|\omega| \geq 2|\boldsymbol{Q}|$. Понеже L е безкраен, знаем, че такава дума съществува. Тогава отново по лемата за покачването, имаме следното разбиване на ω:

$$\omega = xyz,\ |xy| \leq |\boldsymbol{Q}|,\ 1 \leq |y|,\ xz \in L.$$

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

$(2) \Rightarrow (1)$**.** Нека сега $\mathcal{A}$ да разпознава дума ω, за която $|\boldsymbol{Q}| \leq |\omega| < 2|\boldsymbol{Q}|$. Тогава от лемата за покачването следва, че съществува разбиване $\omega = xyz$ 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 3.17*, *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 False
6:     return True
```

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

$$L_1 \subseteq L_2 \;\Leftrightarrow\; (L_1 \setminus L_2) = \emptyset \;\Leftrightarrow\; L_1 \cap \overline{L}_2 = \emptyset.$$

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

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

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

(Проблемът за безкрайност) От *Твърдение 3.18* директно получаваме следния алгоритъм, който разрешава проблемът за безкрайност.

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

3.13 Операции върху езици

Често искаме да получим нов регулярен език, чрез прилагане на някаква операция, като например да вземем префиксите на език, или думите на език, изписани в обратен ред. В този раздел ще разгледаме някои по-сложни операции върху езици и ще видим кои от тях запазват регулярността.

Циклични разместванията

Да разгледаме следната операция върху езици

$$\texttt{Cycle}(L) \stackrel{\text{деф}}{=} \{\omega_2 \cdot \omega_1 \in \mathbf{\Sigma}^\star \ : \ \omega_1 \cdot \omega_2 \in L\}.$$

Следващата задача илюстрира ясно идеята защо понякога е удобно да разбием изчислението на една дума в автомат на части.

Задача 3.9. Докажете, че ако L е регулярен, то $\texttt{Cycle}(L)$ е регулярен.

В [21, стр. 60] е дадено конструктивно решение на тази задача, т.е. в явен вид се строи автомат. Нашият подход е алгебричен, което в случая означава, че представяме $\texttt{Cycle}(L)$ чрез прилагане на *краен* брой регулярни операции върху автоматни езици.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Достатъчно е да проследим еквивалентностите:

$$\begin{aligned}
\omega \in \texttt{Cycle}(L) \ &\Leftrightarrow \ (\exists i)[\, \omega[i:] \cdot \omega[:i] \in L \,] \\
&\Leftrightarrow \ (\exists i)(\exists p \in \boldsymbol{Q})[\, \omega[i:] \in \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \ \& \ \omega[:i] \in \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}) \,] \\
&\Leftrightarrow \ (\exists i)(\exists p \in \boldsymbol{Q})[\, \omega[:i] \cdot \omega[i:] \in \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}) \cdot \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \,] \\
&\Leftrightarrow \ (\exists p \in \boldsymbol{Q})[\, \omega \in \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}) \cdot \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \,] \\
&\Leftrightarrow \ \omega \in \bigcup_{p \in \boldsymbol{Q}} \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}) \cdot \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p).
\end{aligned}$$

Така изразихме $\texttt{Cycle}(L)$ като крайно обединение на регулярни езици. Следователно, $\texttt{Cycle}(L)$ е регулярен език. □

Пермутации на думи

$$\texttt{Perm}(L) = \{\omega \in \mathbf{\Sigma}^\star \ : \ \omega \text{ е пермутация на дума от } L\}.$$

Разполовяване и удвояване на думи

Нека да видим сега следните две операции върху езици:

$$\texttt{Half}(L) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \ : \ \omega \cdot \omega \in L\},$$

$$\texttt{Double}(L) \stackrel{\text{деф}}{=} \{\omega \cdot \omega \in \mathbf{\Sigma}^\star \ : \ \omega \in L\}.$$

На пръв поглед може би изглежда, че тези две операции имат еднакви свойства относно регулярните езици. Оказва се, че това не е така.

Задача 3.10. Докажете, че ако L е регулярен, то $\texttt{Half}(L)$ е регулярен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$, където $\mathcal{A}$ е ДКА. Достатъчно е да проследим веригата от еквивалентности:

$$\begin{aligned}
\omega \in \texttt{Half}(L) &\Leftrightarrow \omega \cdot \omega \in L \\
&\Leftrightarrow (\exists p \in \boldsymbol{Q})[\, \delta^\star(\boldsymbol{s}, \omega) = p \,\&\, \delta^\star(p, \omega) \in \boldsymbol{F} \,] \\
&\Leftrightarrow (\exists p \in \boldsymbol{Q})[\, \omega \in \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \,\&\, \omega \in \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}) \,] \\
&\Leftrightarrow (\exists p \in \boldsymbol{Q})[\, \omega \in \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \cap \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F}) \,] \\
&\Leftrightarrow \omega \in \bigcup_{p \in \boldsymbol{Q}} (\mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \cap \mathcal{L}_{\mathcal{A}}(p, \boldsymbol{F})).
\end{aligned}$$

Така изразихме $\texttt{Half}(L)$ като крайна булева комбинация на регулярни езици. Следователно, $\texttt{Half}(L)$ е регулярен език. □

Задача 3.11. Съществува регулярен език L, за който $\texttt{Double}(L)$ не е регулярен.

Упътване. Достатъчно е да разгледаме $L = \mathcal{L}(\boldsymbol{a}^\star \boldsymbol{b})$. Тогава

$$\texttt{Double}(L) = \{a^n b a^n b \,:\, n \in \mathbb{N}\},$$

за който лесно се съобразява, че не е регулярен. □

Триене на половината дума

Да разгледаме следната операция върху езици:

$$\frac{1}{2}(L) \stackrel{\text{деф}}{=} \{\alpha \in \boldsymbol{\Sigma}^\star \,:\, (\exists \beta \in \boldsymbol{\Sigma}^\star)[\, |\alpha| = |\beta| \,\&\, \alpha\beta \in L \,]\}.$$

Задача 3.12. Докажете, че ако L е регулярен, то $\frac{1}{2}(L)$ е регулярен.

Упътване. Един подход към решаването на тази задача е да изразим езика $\frac{1}{2}(L)$ чрез крайно много регулярни операции приложени върху регулярни езици.

✎ Опитайте се да построите директно автомат за $\frac{1}{2}(L)$. Това е подходът в [21, стр. 58].

Така ще видим, че $\frac{1}{2}(L)$ е регулярен без да строим автомат за него в явен вид. 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.

- $\boldsymbol{Q}_{\mathcal{B}} \stackrel{\text{деф}}{=} \boldsymbol{Q}_{\mathcal{A}}$;
- $\boldsymbol{S}_{\mathcal{B}} \stackrel{\text{деф}}{=} \{\boldsymbol{s}_{\mathcal{A}}\}$;
- $\boldsymbol{F}_{\mathcal{B}} \stackrel{\text{деф}}{=} \boldsymbol{F}_{\mathcal{A}}$;
- $\boldsymbol{\Delta}_{\mathcal{B}}(q, x) \stackrel{\text{деф}}{=} \{\boldsymbol{\delta}_{\mathcal{A}}(q, y) \,:\, y \in \boldsymbol{\Sigma}\}$, за произволни $q \in \boldsymbol{Q}$ и $x \in \boldsymbol{\Sigma}$

Понеже $\boldsymbol{Q}_{\mathcal{A}} = \boldsymbol{Q}_{\mathcal{B}}$, ще пишем $\boldsymbol{Q}$.

Тогава $\beta \in \mathcal{L}(\mathcal{B}) \;\Leftrightarrow\; (\exists \alpha \in \boldsymbol{\Sigma}^\star)[|\alpha| = |\beta| \;\&\; \alpha \in \mathcal{L}(\mathcal{A})]$.

$$\begin{aligned} \alpha \in \frac{1}{2}(L) \;&\Leftrightarrow\; (\exists p \in \boldsymbol{Q})[\alpha \in \mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \;\&\; \alpha \in \mathcal{L}_{\mathcal{B}}(p, \boldsymbol{F})] \\ &\Leftrightarrow\; \alpha \in \bigcup_{p \in \boldsymbol{Q}} (\mathcal{L}_{\mathcal{A}}(\boldsymbol{s}, p) \cap \mathcal{L}_{\mathcal{B}}(p, \boldsymbol{F})). \end{aligned}$$

□

Разбиване на дума на три части

Да разгледаме следните операции върху езици.

$$\texttt{Triples}_2(L) \stackrel{\text{деф}}{=} \{\omega_2 \in \boldsymbol{\Sigma}^\star \,:\, (\exists\omega_1)(\exists\omega_3)[\,\omega_1\omega_2\omega_3 \in L \;\&\; |\omega_1| = |\omega_2| = |\omega_3|\,]\,\};$$
$$\texttt{Triples}_{1,3}(L) \stackrel{\text{деф}}{=} \{\omega_1\omega_3 \in \boldsymbol{\Sigma}^\star \,:\, (\exists\omega_2)[\,\omega_1\omega_2\omega_3 \in L \;\&\; |\omega_1| = |\omega_2| = |\omega_3|\,]\,\}.$$

Отново се оказва, че тези две операции са принципно различни относно регулярните езици.

Задача 3.13. Докажете, че ако L е регулярен език, то $\texttt{Triples}_2(L)$ е регулярен.

Упътване. Използвайки означенията от *Задача 3.12*, можем да изразим езика така:

$$\texttt{Triples}_2(L) = \bigcup_{p,q \in \boldsymbol{Q}} (\mathcal{L}_{\mathcal{B}}(\boldsymbol{s}, p) \cap \mathcal{L}_{\mathcal{A}}(p, q) \cap \mathcal{L}_{\mathcal{B}}(q, \boldsymbol{F})).$$

□

Задача 3.14. Докажете, че съществува регулярен език L, за който $\texttt{Triples}_{1,3}(L)$ *не* е регулярен.

Упътване. Разгледайте езика $L = \mathcal{L}(\boldsymbol{a}^+\boldsymbol{b}\boldsymbol{a}^+\boldsymbol{b}\boldsymbol{a}^+)$. Ако допуснем, че езикът $\texttt{Triples}_{1,3}(L)$ е регулярен, то

Интересно е, че за регулярен език L, $\texttt{Triples}_{1,3}(L)$ е безконтекстен.
Вижте *Задача 4.23*.

$$M \stackrel{\text{деф}}{=} \texttt{Triples}_{1,3}(L) \cap \mathcal{L}(\boldsymbol{a}^+\boldsymbol{b}\boldsymbol{b}\boldsymbol{a}^+)$$

също би трябвало да е регулярен, но лесно се съобразява, че езикът

$$M = \{a^n bba^n \ : \ n \geq 1\},$$

който не е регулярен. □

Сливане на думи

Да дефинираме рекурсивно функцията $\texttt{merge} : \mathbf{\Sigma}^\star \times \mathbf{\Sigma}^\star \to \mathcal{P}(\mathbf{\Sigma}^\star)$, която *смесва* две думи по всички възможни начини така:

$$\texttt{merge}(\alpha, \varepsilon) \stackrel{\text{деф}}{=} \{\alpha\};$$
$$\texttt{merge}(\varepsilon, \beta) \stackrel{\text{деф}}{=} \{\beta\};$$
$$\texttt{merge}(a \cdot \alpha, b \cdot \beta) \stackrel{\text{деф}}{=} \{a\} \cdot \texttt{merge}(\alpha, b \cdot \beta) \cup \{b\} \cdot \texttt{merge}(a \cdot \alpha, \beta).$$

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 3.15. Ако L_1 и L_2 са регулярни езици, то $\texttt{Merge}(L_1, L_2)$ е регулярен.

3.14 Матрично представяне на автомат (*)

Булеви матрици

Нека тук за простота да интерпретираме 1 като булевата стойност „истина” и 0 като булевата стойност „неистина”. Да положим $\texttt{Bool} = \{0, 1\}$ и да означим с $M_{n\times n}(\texttt{Bool})$ класът от булеви матрици с размерност $n \times n$.

Знаем, че всеки насочен граф $G = (V, E)$ може да се представи с неговата матрица на съседство $\boldsymbol{M}$, където $\boldsymbol{M}\,[i]\,[j] = 1$ точно тогава, когато $(v_i, v_j) \in E$. Ясно е, че $M^k\,[i]\,[j] = 1$ точно тогава, когато съществува път с дължина k от v_i до v_j в графа G. Нека е даден автомат $\mathcal{A} = \langle \boldsymbol{\Sigma}, \boldsymbol{Q}, \boldsymbol{s}, \boldsymbol{\delta}, \boldsymbol{F}\rangle$ като $\boldsymbol{Q} = \{q_0, \ldots, q_{n-1}\}$. Дефинираме булевата матрица $\boldsymbol{M}_a \in M_{n\times n}(\texttt{Bool})$ по следния начин:

$$\boldsymbol{M}_a\,[q]\,[p] = 1 \;\Leftrightarrow\; \boldsymbol{\delta}(q, a) = p.$$

Когато трябва изрично да подчертаем за кой автомат говорим, ще пишем $\boldsymbol{M}_{\mathcal{A}}$ и $\boldsymbol{M}_{\mathcal{A},\omega}$.

Когато нямаме индекс за буква a, то това ще означава, че не се интересуваме от буквата, за която правим прехода и тогава дефинираме

$$\boldsymbol{M}\,[q]\,[p] = 1 \;\Leftrightarrow\; (\exists a \in \boldsymbol{\Sigma})[\,\boldsymbol{\delta}(q, a) = p\,].$$

Нека положим $\boldsymbol{I}$ да бъде матрицата идентитет. Индуктивно дефинираме $\boldsymbol{M}_\omega$ за всяка дума $\omega \in \boldsymbol{\Sigma}^\star$.

- $\boldsymbol{M}_\varepsilon \stackrel{\text{деф}}{=} \boldsymbol{I}$.
- $\boldsymbol{M}_{\beta a} \stackrel{\text{деф}}{=} \boldsymbol{M}_\beta \cdot \boldsymbol{M}_a$.

Твърдение 3.19. $\boldsymbol{M}_\omega\,[q]\,[p] = 1$ точно тогава когато $\boldsymbol{\delta}^\star(q, \omega) = p$.

Упътване. Индукция по $|\omega|$. □

✍ Съобразете, че може да дефинирате $\mathcal{M}_{\mathcal{A}}$ по сходен начин и за НКА $\mathcal{A}$. Така ще получите ново доказателство, че езикът на всеки НКА се разпонава и от ДКА.

При даден краен автомат $\mathcal{A}$ с n състояния, можем да дефинираме ДКА $\mathcal{M}_{\mathcal{A}}$ по следния начин:

- Състоянията на $\mathcal{M}_{\mathcal{A}}$ са елементите на $M_{n\times n}(\texttt{Bool})$;
- Началното състояние на новия автомат е матрицата идентитет $\boldsymbol{I}$;
- Функцията на преходите е дефинирана като $\boldsymbol{\delta}(\boldsymbol{K}, a) \stackrel{\text{деф}}{=} \boldsymbol{K} \cdot \boldsymbol{M}_{\mathcal{A},a}$;
- $\boldsymbol{F} \stackrel{\text{деф}}{=} \{\boldsymbol{K} \,:\, \boldsymbol{K}\,[\boldsymbol{s}]\,[q] = 1 \text{ за някое } q \in \boldsymbol{F}_{\mathcal{A}}\}$.

Твърдение 3.20. За всеки краен автомат $\mathcal{A}$ е изпълнено, че

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{M}_{\mathcal{A}}).$$

Да напомним, че в *Задача 3.10* вече сме разглеждали операцията

$$\texttt{Half}(L) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \,:\, \omega \cdot \omega \in L\}.$$

Задача 3.16. Ако L е автоматен език, то $\texttt{Half}(L)$ е автоматен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Разгледайте автомат $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{M}_\mathcal{A} \times \mathcal{M}_\mathcal{A}$, където

$$\boldsymbol{F}_\mathcal{B} \stackrel{\text{деф}}{=} \{\langle \boldsymbol{K}_1, \boldsymbol{K}_2\rangle \,:\, (\boldsymbol{K}_1 \cdot \boldsymbol{K}_2)[\boldsymbol{s}][q] = 1 \text{ за някое } q \in \boldsymbol{F}_\mathcal{A}\}.$$

□

Да напомним, че в *Задача 3.12* вече сме разглеждали операцията

$$\tfrac{1}{2}(L) \stackrel{\text{деф}}{=} \{\alpha \,:\, (\exists \beta \in \mathbf{\Sigma}^{|\alpha|})[\alpha\beta \in L]\}.$$

Задача 3.17. Докажете, че ако L е автоматен, то $\frac{1}{2}(L)$ е автоматен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$, където $\mathcal{A}$ е краен автомат. Разгледайте автомат $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{A} \times \mathcal{M}_\mathcal{A}$, където

$$\delta_\mathcal{B}(\langle q, \boldsymbol{K}\rangle, a) \stackrel{\text{деф}}{=} \langle \delta(q,a), \boldsymbol{K} \cdot \boldsymbol{M}_\mathcal{A}\rangle;$$
$$\boldsymbol{F}_\mathcal{B} \stackrel{\text{деф}}{=} \{\langle q, \boldsymbol{K}\rangle \,:\, \boldsymbol{K}[q][p] = 1 \text{ за някое } p \in \boldsymbol{F}_\mathcal{A}\}.$$

□

Да припомним следната операция върху езици

$$\texttt{Triples}_2(L) \stackrel{\text{деф}}{=} \{\omega_2 \in \mathbf{\Sigma}^\star \,:\, (\exists\omega_1)(\exists\omega_3)[\,\omega_1\omega_2\omega_3 \in L \,\&\, |\omega_1| = |\omega_2| = |\omega_3|\,]\,\}.$$

Задача 3.18. Докажете, че ако L е регулярен, то $\texttt{Triples}_2(L)$ е регулярен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Дефинираме $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{M}_\mathcal{A} \times \mathcal{M}_\mathcal{A} \times \mathcal{M}_\mathcal{A}$.

$$\boldsymbol{\delta}_\mathcal{B}(\langle \boldsymbol{K}_1, \boldsymbol{K}_2, \boldsymbol{K}_3\rangle, a) \stackrel{\text{деф}}{=} \langle \boldsymbol{K}_1 \cdot \boldsymbol{M}_\mathcal{A}, \boldsymbol{K}_2 \cdot \boldsymbol{M}_{\mathcal{A},a}, \boldsymbol{K}_3 \cdot \boldsymbol{M}_\mathcal{A}\rangle;$$
$$\boldsymbol{F}_\mathcal{B} \stackrel{\text{деф}}{=} \{\langle \boldsymbol{K}_1, \boldsymbol{K}_2, \boldsymbol{K}_3\rangle \,:\, (\boldsymbol{K}_1 \cdot \boldsymbol{K}_2 \cdot \boldsymbol{K}_3)[\boldsymbol{s}][p] = 1 \text{ за някое } p \in \boldsymbol{F}_\mathcal{A}\}.$$

□

Да дефинираме следната операция върху езици:

$$\texttt{Repeat}(L) \stackrel{\text{деф}}{=} \{\omega\sharp a^n \,:\, \omega^n \in L\}.$$

Задача 3.19. Докажете, че ако L е автоматен, то $\texttt{Repeat}(L)$ е автоматен.

Да разгледаме следната операция върху езици:

$$\sqrt{L} \stackrel{\text{деф}}{=} \{\alpha \in \mathbf{\Sigma}^\star \,:\, (\exists \beta \in \mathbf{\Sigma}^\star)[\, |\beta| = |\alpha|^2 \,\&\, \alpha \cdot \beta \in L\,]\}.$$

Задача 3.20. Ако L е автоматен, то $\sqrt{L}$ е автоматен.

Използваме, че $(x+1)^2 = x^2 + 2x + 1$.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Да разгледаме автомат $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{A} \times \mathcal{M}_\mathcal{A} \times \mathcal{M}_\mathcal{A}$, където

$$\boldsymbol{\delta}_\mathcal{B}(\langle q, \boldsymbol{K}_1, \boldsymbol{K}_2\rangle, a) \stackrel{\text{деф}}{=} \langle \boldsymbol{\delta}_\mathcal{A}(q,a), \boldsymbol{K}_1 \cdot \boldsymbol{K}_2 \cdot \boldsymbol{M}_\mathcal{A}, \boldsymbol{K}_2 \cdot \boldsymbol{M}_\mathcal{A} \cdot \boldsymbol{M}_\mathcal{A}\rangle;$$
$$\boldsymbol{F} \stackrel{\text{деф}}{=} \{\langle q, \boldsymbol{K}_1, \boldsymbol{K}_2\rangle \,:\, \boldsymbol{K}_1\,[q]\,[p] = 1 \text{ за някое } p \in \boldsymbol{F}_\mathcal{A}\}.$$

□

Да разгледаме следната операция върху езици:

$$\log(L) \stackrel{\text{деф}}{=} \{\alpha \in \mathbf{\Sigma}^\star \,:\, (\exists \beta \in \mathbf{\Sigma}^\star)[\, |\beta| = 2^{|\alpha|} \,\&\, \alpha \cdot \beta \in L\,]\}.$$

Твърдение 3.21. Ако L е автоматен, то $\log(L)$ е автоматен.

Използваме, че $2^{n+1} = 2^n + 2^n$.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Да разгледаме автомата $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{A} \times \mathcal{M}_\mathcal{A}$, където

$$\boldsymbol{s}_\mathcal{B} \stackrel{\text{деф}}{=} \langle \boldsymbol{s}, \boldsymbol{M}_\mathcal{A}\rangle;$$
$$\boldsymbol{\delta}_\mathcal{B}(\langle q, \boldsymbol{K}\rangle, a) \stackrel{\text{деф}}{=} \langle \boldsymbol{\delta}_\mathcal{A}(q,a), \boldsymbol{K} \cdot \boldsymbol{K}\rangle;$$
$$\boldsymbol{F}_\mathcal{B} \stackrel{\text{деф}}{=} \{\langle q, \boldsymbol{K}\rangle \,:\, \boldsymbol{K}\,[q]\,[p] = 1 \text{ за някое } p \in \boldsymbol{F}_\mathcal{A}\}.$$

□

Да разгледаме следната операция върху езици:

$$\texttt{root}(L) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \,:\, \omega^{|\omega|} \in L\}.$$

Задача 3.21. Ако L е автоматен, то $\texttt{root}(L)$ е автоматен.

Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Да разгледаме $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{M}_\mathcal{A} \times \mathcal{M}_\mathcal{A} \times \mathcal{M}_\mathcal{A}$, който ще бъде НКА, където

$$\boldsymbol{S}_\mathcal{B} \stackrel{\text{деф}}{=} \{\langle \boldsymbol{I}, \boldsymbol{K}, \boldsymbol{I}\rangle \,:\, \boldsymbol{K} \text{ е булева матрица}\};$$

$$\boldsymbol{\Delta}_\mathcal{B}(\langle \boldsymbol{K}_1, \boldsymbol{K}_2, \boldsymbol{K}_3\rangle, a) \stackrel{\text{деф}}{=} \{\langle \boldsymbol{K}_1 \cdot \boldsymbol{M}_{\mathcal{A},a}, \boldsymbol{K}_2, \boldsymbol{K}_3 \cdot \boldsymbol{K}_2\rangle\};$$

$$\boldsymbol{F}_\mathcal{B} \stackrel{\text{деф}}{=} \{\langle \boldsymbol{K}_1, \boldsymbol{K}_2, \boldsymbol{K}_3\rangle \,:\, \boldsymbol{K}_1 = \boldsymbol{K}_2 \,\&\, \boldsymbol{K}_3\,[\boldsymbol{s}]\,[p] = 1 \text{ за някое } p \in \boldsymbol{F}_\mathcal{A}\}.$$

Проверете, че е изпълнено свойството:

$$(\langle \boldsymbol{I}, \boldsymbol{M}_\omega, \boldsymbol{I}\rangle, \omega) \vdash^\star_\mathcal{B} (\langle \boldsymbol{M}_\omega, \boldsymbol{M}_\omega, \boldsymbol{M}_\omega^{|\omega|}\rangle, \varepsilon).$$

□

За една булева матрица $\boldsymbol{M}$, да дефинираме $\boldsymbol{M}^\star$ по следния начин:

$$\boldsymbol{M}^\star = \boldsymbol{I} + \boldsymbol{M} + \boldsymbol{M}^2 + \boldsymbol{M}^3 + \cdots$$

или с други думи

$$\boldsymbol{M}^\star = \sum_{n=0}^{\infty} \boldsymbol{M}^n.$$

Нека се ограничим до булеви матрици с фиксирана размерност k. Понеже можем да интерпретираме $\boldsymbol{M}^n\,[q]\,[p] = 1$ като съществуването на път от q до p с дължина n, то е ясно, че за $n > k$, то имаме вече цикъл в пътя и следователно $\boldsymbol{M}^k = \boldsymbol{M}^{k+1}$. Оттук $(\forall m \geq k)[\boldsymbol{M}^k = \boldsymbol{M}^m]$. Така получаваме, че

$$\boldsymbol{M}^\star = \sum_{n=0}^{k} \boldsymbol{M}^n.$$

Да разгледаме операцията

$$\texttt{unstar}(L) \stackrel{\text{деф}}{=} \{\omega \in \boldsymbol{\Sigma}^\star \,:\, \omega^\star \in L\}.$$

Задача 3.22. Докажете, че ако L е автоматен, $\texttt{unstar}(L)$ е автоматен.

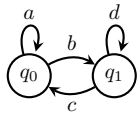
Упътване. Нека $L = \mathcal{L}(\mathcal{A})$. Разгледайте $\mathcal{B} \stackrel{\text{деф}}{=} \mathcal{M}_\mathcal{A}$, където

$$\boldsymbol{F}_\mathcal{B} \stackrel{\text{деф}}{=} \{\boldsymbol{K} \,:\, \boldsymbol{K}^\star\,[\boldsymbol{s}]\,[p] = 1 \text{ за някое } p \in \boldsymbol{F}_\mathcal{A}\}.$$

□

Матрици от регулярни изрази

Булевите матрици са полезни, когато се интересуваме от въпроса за достижимост, например, има ли път с думата ω от q до p в автомата $\mathcal{A}$. Нека сега да разгледаме по-общия вариант, където се интересуваме от това да опишем всички пътища между две състояния в автомат. Ще разгледаме само случая за матрици 2×2, но нашите разглеждания могат да се обобщят. Да разгледаме матрицата



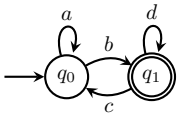
(а) Матрицата M представя функцията на преходите на автомата $\mathcal{A}$. Тогава $M^\star$ описва пътищата между произволна двойка състояния на автомата.

$$M \stackrel{\text{деф}}{=} \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Как можем да намерим $M^\star$? Знаем, че $M^\star$ е *единственото* решение на уравнението от матрици

$$X = M \cdot X + I,$$

където $X = \begin{bmatrix} x & y \\ u & v \end{bmatrix}$ и $I = \begin{bmatrix} \varepsilon & \emptyset \\ \emptyset & \varepsilon \end{bmatrix}$.



(б) Езикът на автомата се описва с регулярния език:

$$\begin{bmatrix} \varepsilon & \emptyset \end{bmatrix} \cdot M^\star \cdot \begin{bmatrix} \emptyset \\ \varepsilon \end{bmatrix}$$

Това означава, че трябва да решим уравниението от матрици:

$$\begin{bmatrix} x & y \\ u & v \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} x & y \\ u & v \end{bmatrix} + \begin{bmatrix} \varepsilon & \emptyset \\ \emptyset & \varepsilon \end{bmatrix},$$

което означава да решим системата:

$$\left| \begin{array}{rcl} x & = & ax + bu + \varepsilon \\ y & = & ay + bv + \emptyset \\ u & = & cx + du + \emptyset \\ v & = & cy + dv + \varepsilon. \end{array} \right.$$

Използвайки *Лема 2.2*, получаваме следното:

$$\begin{aligned} y &= a^\star bv \\ u &= d^\star cx. \end{aligned}$$

Заместваме и получаваме:

$$\begin{aligned} v &= ca^\star bv + dv + \varepsilon \\ &= (ca^\star b + d)v + \varepsilon \\ &= (ca^\star b + d)^\star; \\ x &= ax + bd^\star cx + \varepsilon \\ &= (a + bd^\star c)x + \varepsilon \\ &= (a + bd^\star c)^\star. \end{aligned}$$

Накрая получаваме:

$$\left|\begin{array}{lcl} \boldsymbol{x} & = & (\boldsymbol{a}+\boldsymbol{b}\boldsymbol{d}^\star\boldsymbol{c})^\star \\ \boldsymbol{y} & = & \boldsymbol{a}^\star\boldsymbol{b}(\boldsymbol{c}\boldsymbol{a}^\star\boldsymbol{b}+\boldsymbol{d})^\star \\ \boldsymbol{u} & = & \boldsymbol{d}^\star\boldsymbol{c}(\boldsymbol{a}+\boldsymbol{b}\boldsymbol{d}^\star\boldsymbol{c})^\star \\ \boldsymbol{v} & = & (\boldsymbol{c}\boldsymbol{a}^\star\boldsymbol{b}+\boldsymbol{d})^\star. \end{array}\right.$$

Ясно е, че $\boldsymbol{M}^\star$ задава пълното описание на автомата. Сега, ако фиксираме q_0 да е началното състояние и q_1 да бъде финалното, то можем да намерим регулярния израз за $\mathcal{L}(\mathcal{A})$ по следния начин:

$$\begin{bmatrix} \varepsilon & \emptyset \end{bmatrix} \cdot \boldsymbol{M}^\star \cdot \begin{bmatrix} \emptyset \\ \varepsilon \end{bmatrix} = \boldsymbol{a}^\star\boldsymbol{b}(\boldsymbol{c}\boldsymbol{a}^\star\boldsymbol{b}+\boldsymbol{d})^\star.$$

3.15 Задачи

Задача 3.23. За всеки от следните езици L, постройте минимален краен детерминиран автомат $\mathcal{A}$, който разпознава езика L, където:

а) $L = \{a^n b : n \geq 0\}$;

б) $L = \{a, b\}^\star \setminus \{\varepsilon\}$;

в) $L = \{\omega \in \{a, b\}^\star : |\omega|_a$ и $|\omega|_b$ са четни$\}$;

г) $L = \{a^n b^m : n, m \geq 1\}$;

д) $L = \{a, b\}^\star \setminus \{a\}$;

е) $L = \{\omega \in \{a, b\}^\star : |\omega|_{ab} \equiv 1 \bmod 2\}$;

ж) $L = \{\omega \in \{a, b\}^\star : |\omega|_a$ е четно $\& \, |\omega|_b \leq 1\}$;

з) $L = \{\omega \in \{a, b\}^\star : |\omega| \leq 3\}$;

и) $L = \{\omega \in \{a, b\}^\star : \omega$ не започва с $ab\}$;

к) $L = \{\omega \in \{a, b\}^\star : \omega$ завършва с ab или ba$\}$;

л) $L = \{\omega \in \{a, b\}^\star : |\omega| \equiv 0 \pmod 2 \ \& \ |\omega|_a = 1\}$;

м) $L = \{\omega \in \{a, b\}^\star : |\omega|_a \equiv 0 \bmod 3 \ \& \ |\omega|_b \equiv 1 \bmod 2\}$;

н) $L = \{\omega \in \{a, b\}^\star : |\omega|_a \equiv 0 \bmod 2 \ \vee \ |\omega|_b = 2\}$;

о) $L = \{\omega \in \{a, b\}^\star : |\omega|_{ab} = |\omega|_{ba}\}$;

Задача 3.24. Нека $\Sigma = \{a, b\}$. Проверете дали L е регулярен, където

а) $L = \{a^i b^i : i \in \mathbb{N}\}$;

б) $L = \{a^i b^i : i, j \in \mathbb{N} \ \& \ i \neq j\}$;

в) $L = \{a^i b^j : i > j\}$;

г) $L = \{a^n b^m : n$ дели $m\}$.

д) $L = \{a^{2^n} : n \geq 1\}$;

е) $L = \{a^m b^n a^{m+n} : m \geq 1 \ \& \ n \geq 1\}$;

ж) $L = \{a^{n \cdot m} : n, m$ са прости числа$\}$;

з) $L = \{\omega \in \{a, b\}^\star : |\omega|_a = |\omega|_b\}$;

и) $L = \{\omega\omega : \omega \in \{a, b\}^\star\}$;

к) $L = \{\omega\omega^{\mathtt{rev}} : \omega \in \{a, b\}^\star\}$;

л) $L = \{\alpha\beta\beta \in \{a, b\}^\star : \beta \neq \varepsilon\}$;

м) $L = \{a^{2^n} : n \geq 0\}$;

н) $L = \{a^m b^n : n \neq m\}$;

о) $L = \{\omega \in \Sigma^\star : | \, |\omega|_a - |\omega|_b \, | \leq 2\}$;

п) $L = \{\alpha\beta\alpha : \alpha, \beta \in \Sigma^\star \ \& \ |\beta| \leq |\alpha|\}$;

р) $L = \{\beta\gamma\gamma^{\mathtt{rev}} : \beta, \gamma \in \Sigma^\star \ \& \ |\beta| \leq |\gamma|\}$;

с) $L = \{c^k a^n b^m : k, m, n > 0 \ \& \ n \neq m\}$;

т) $L = \{\omega \in \{a, b\}^\star : |\omega|_a$ не дели $|\omega|_b\}$;

у) $L = \{\omega \in \{a, b\}^\star : |\omega|_a = 2|\omega|_b\}$;

ф) $L = \{\omega \in \{a, b\}^\star : | \, |\omega|_a - |\omega|_b \, | \leq 3\}$;

х) $L = \{\alpha \in \{a, b\}^\star :$ за всяка представка ω на $\alpha, | \, |\omega|_a - |\omega|_b \, | \leq 2\}$;

ц) $L = \{a^{f_n} : f_0 = f_1 = 1 \ \& \ f_{n+2} = f_{n+1} + f_n\}$.

Задача 3.25. 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?

б) Вярно ли е, че ако L_1 и L_2 са регулярни езици, то $L_1 \oplus L_2$ е регулярен език?

Задача 3.26. 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 3.27. Нека $\Sigma = \{a, b, c\}$ и да разгледаме следната операция [7, стр. 88].

$$\texttt{Sort} : \mathcal{P}(\Sigma^\star) \to \mathcal{P}(\Sigma^\star),$$

където

$$\texttt{Sort}(L) \stackrel{\text{деф}}{=} \{a^{|\omega|_a} b^{|\omega|_b} c^{|\omega|_c} \mid \omega \in L\}.$$

Вярно ли е, че ако L е регулярен, то $\texttt{Sort}(L)$ е регулярен ?

Задача 3.28. Докажете, че ако L е автоматен език, то $\texttt{Pref}(L)$ е автоматен.

Задача 3.29. Докажете, че ако L е автоматен език, то $\texttt{Suff}(L)$ е автоматен.

Задача 3.30. Докажете, че ако L е автоматен език, то $\texttt{Rev}(L)$ е автоматен.

Задача 3.31. Докажете, че ако L е автоматен език, то $\alpha^{-1}(L)$ е автоматен.

Задача 3.32. Докажете, че ако L е автоматен език и M е произволен език, то $M^{-1}(L)$ е автоматен.

Задача 3.33. Вярно ли е, че за всеки недетерминиран краен автомат $\mathcal{N}$ съществува недетерминиран краен автомат $\mathcal{N}'$ с *едно финално* състояние, за който $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{N}')$?

Задача 3.34. Вярно ли е, че за всеки детерминиран краен автомат $\mathcal{A}$, съществува детерминиран краен автомат $\mathcal{A}'$ с *едно финално* състояние, за който $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$?

Глава 4

Безконтекстни езици и стекови автомати

Видяхме, че има много прости езици, например $\{a^n b^n \ : \ n \in \mathbb{N}\}$, които не са регулярни. В тази глава ще разгледаме един по-широк клас от езици, така наречените безконтекстни езици, които се характеризират като решения на системи от по-общ тип в сравнение със системите за регулярните езици.

4.1 Извод върху безконтекстна граматика

Безконтекстна граматика е наредена четворка от вида $G \stackrel{\text{деф}}{=} \langle \boldsymbol{V}, \boldsymbol{\Sigma}, \boldsymbol{R}, \boldsymbol{S} \rangle$, където:

- $\boldsymbol{V}$ е крайно множество от *променливи* (нетерминали);
- $\boldsymbol{\Sigma}$ е крайно множество от *букви* (терминали), като $\boldsymbol{\Sigma} \cap \boldsymbol{V} = \emptyset$;
- $\boldsymbol{R} \subseteq \boldsymbol{V} \times (\boldsymbol{V} \cup \boldsymbol{\Sigma})^\star$ е крайно множество от *правила*. За по-добра яснота, обикновено правилата $(A, \beta) \in \boldsymbol{R}$ ще означаваме като $A \to_G \beta$. Когато е ясно за коя граматика говорим, ще пишем просто $A \to \beta$.
- $\boldsymbol{S} \in \boldsymbol{V}$ е началната променлива (нетерминал).

В [10] правилата се наричат *productions* или *production rules*.

При автоматите, ние разпознавахме думи. При граматиките, ние ще *генерираме* думи. Например, да разгледаме граматиката G с правила

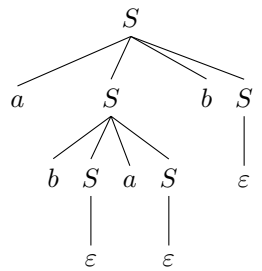
$$S \to aSbS, \quad S \to bSaS \quad \text{и} \quad S \to \varepsilon.$$

Обикновено ще записваме по-компактно правилата на G така:

$$S \to aSbS \mid bSaS \mid \varepsilon.$$

Тук може би е по-добре да се използва + вместо | и така дясната страна на правилата ще отговарят на регулярни изрази.

Думите, генерирани от граматиката са тези думи, при които започваме от променливата S и след крайно заместване на променлива с дясната страна на правило, достигаме до дума без променливи. Например, да видим как можем да генерираме думата $abab$.



Фигура 4.1: Синтактично дърво, което показва как граматиката G генерира думата $abab$.

(1) Започваме от началната променлива S. В конкретния случай, това е и единствената променлива в граматиката.

(2) Заместваме S с $aSbS$. Ще казваме, че сме приложили правилото $S \to aSbS$.

(3) В междинната дума $aSbS$ имаме две срещания на променливата S. Нека да заместим първото срещане на S с $bSaS$ и второто срещане с ε, т.е. прилагаме правилата $S \to bSaS$ и правилото $S \to \varepsilon$.

(4) Получаваме думата $abSaSb$. Вече е ясно, че прилагаме два пъти $S \to \varepsilon$ и получаваме окончателно думата $abab$.

Ние се интересуваме най-вече от крайния резултат, а не от редицата от правила, които прилагаме. Например, на стъпка (3) няма значение дали първо сме приложили $S \to aSbS$ или $S \to \varepsilon$. Резултатът е все същия - думата $abSab$. Поради тази причина, най-добре се илюстрира прилагането на правилата като разглеждаме дърво, чиито върхове са елементи на $\boldsymbol{V} \cup \boldsymbol{\Sigma} \cup \{\varepsilon\}$ и закономерността между

върховете и техните наследници се определя от правилата на граматиката. За конкретния пример получаваме дървото на *Фигура 4.1*.

Първата ни задача е систематично да изучим как можем да генерираме думи от граматика. За произволна безконтекстна граматика G, дефинираме релацията $X \overset{\ell}{\lhd} \alpha$, където $X \in \boldsymbol{V} \cup \boldsymbol{\Sigma}$ и $\alpha \in (\boldsymbol{V} \cup \boldsymbol{\Sigma})^\star$. Едновременно с това ще конструираме дърво съответстващо на този извод. Ще наричаме тези дървета *синтактични*, защото върховете на тези дървета ще бъдат елементи на $\boldsymbol{\Sigma} \cup \boldsymbol{V} \cup \{\varepsilon\}$.

(1) Да започнем с базовия случай. Имаме следното правило:

$$\frac{X \in \boldsymbol{V} \cup \boldsymbol{\Sigma}}{X \overset{0}{\lhd} X}$$

На този извод съответства синтактично дърво с височина 0, чийто единствен връх е означен с X. Този връх играе ролята на корен и на единствено листо на дървото. Това дърво е изобразено на *Фигура 4.2а*.

(2) Да разгледаме правило $X \to_G X_1 \cdots X_n$. Тук имаме два случая в зависимост от стойността на n. Нека първо $n \geq 1$. Тогава имаме правилото:

$$\frac{X \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \omega_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \omega_n}{X \overset{\ell+1}{\lhd} \omega_1 \cdots \omega_n} \ (\ell = \sup\{\ell_1, \ldots, \ell_n\})$$

Нека на изводите $X_i \overset{\ell_i}{\lhd} \omega_i$ да съответстват синтактични дървета с корени X_i, листа ω_i и височина ℓ_i. Тогава на извода $X \overset{\ell}{\lhd} \omega_1 \cdots \omega_n$ съответства синтактично дърво с корен X, листа $\omega_1, \ldots, \omega_n$, височина ℓ, като директните наследници на X са възлите $X_1, \ldots, X_n$. Това синтактично дърво е изобразено на *Фигура 4.2б*.

(3) Остана да разгледаме случая, когато $X \to_G X_1 \cdots X_n$ за $n = 0$. Тогава, по дефиниция, $X_1 \cdots X_n = \varepsilon$. Освен това, понеже $\sup(U)$ означава най-малкото естествено число по-голямо от всички елементи на U, то $\sup(\emptyset) = 0$. Това означава, че имаме следното правило като частен случай на (2):

$$\frac{X \to_G \varepsilon}{X \overset{1}{\lhd} \varepsilon}$$

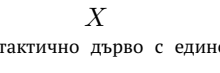
На този извод съответства синтактично дърво с корен X, единствено листо ε и височина 1. Това дърво е изобразено на *Фигура 4.2в*.

Когато не се интересуваме от дължината на извода, ще пишем $A \lhd \omega$, т.е.

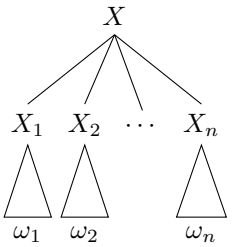
$$A \lhd \omega \ \Leftrightarrow \ (\exists \ell)[\, A \overset{\ell}{\lhd} \omega \,].$$

Сега вече сме готови да дефинираме официално какво е език на граматика.

Фигура 4.2: Синтактични дървета съответстващи на извод в граматика:



(а) Синтактично дърво с единствен връх X.



(б) Синтактично дърво с корен X, чиито преки наследници са $X_1, \ldots, X_n$ и листата формират думата $\omega_1 \cdots \omega_n$.

(в) Синтактично дърво с корен X и единствен наследник ε.

Да напомним дефиницията

$$\sup(U) = \min\{m : (\forall u \in U)[u \leq m]\}$$

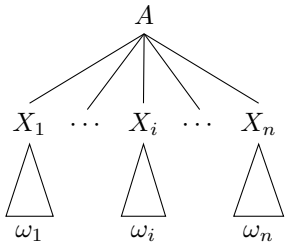
Определение 4.1. Нека G е произволна безконтекстна граматика. Тогава дефинираме езикът на G да бъде

$$\mathcal{L}(G) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \,:\, S \lhd \omega\}.$$

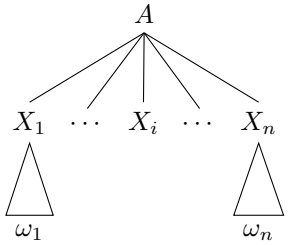
Ще продължим с разглеждането на някои свойства на релацията за извод в граматика, които ще ни бъдат полезни по-късно.

Твърдение 4.1. Нека $A \overset{\ell+1}{\lhd} \omega$ като $\ell \geq 1$. Тогава съществува разбиване на ω на три части като $\omega = \alpha\beta\gamma$ и съществува променлива B, за които

$$A \overset{\leq \ell+1}{\lhd} \alpha B \gamma \text{ и } B \overset{\ell}{\lhd} \beta.$$



(а) Синтактично дърво, което илюстрира извода $A \overset{\ell+1}{\lhd} \omega$, където $X_i \overset{\ell}{\lhd} \omega_i$.



(б) Синтактично дърво, което илюстрира извода $A \overset{m+1}{\lhd} \alpha X_i \gamma$, където $m \leq \ell$.

Доказателство. Ясно е, че изводът $A \overset{\ell+1}{\lhd} \omega$ е получен чрез прилагане на правило (2):

$$\frac{A \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \omega_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \omega_n}{A \overset{\ell+1}{\lhd} \underbrace{\omega_1 \cdots \omega_n}_{\omega}} \; (\ell = \sup\{\ell_1, \ldots, \ell_n\})$$

Щом $\ell = \sup\{\ell_1, \ldots, \ell_n\}$, то съществува поне един индекс i, за който $\ell = \ell_i$. Нека $B \stackrel{\text{деф}}{=} X_i$ за един такъв индекс i и да разбием ω по следния начин на три части:

$$\underbrace{\omega_1 \cdots \omega_{i-1}}_{\alpha} \cdot \underbrace{\omega_i}_{\beta} \cdot \underbrace{\omega_{i+1} \cdots \omega_n}_{\gamma}.$$

Тогава получаваме следния извод като приложим правилото (2) така:

$$\frac{A \to_G X_1 \cdots X_n \qquad X_j \overset{\ell_j}{\lhd} \omega_j \text{ за } j \neq i \qquad X_i \overset{0}{\lhd} X_i}{A \overset{m+1}{\lhd} \alpha X_i \gamma} \; (m = \sup\{\ell_j \,:\, j \neq i\})$$

Ясно е, че $m \leq \ell$. Заключаваме, че $A \overset{\leq \ell+1}{\lhd} \alpha X_i \gamma$ и $B_i \overset{\ell}{\lhd} \beta$. □

Прилагайки *Твърдение 4.1* краен брой пъти, получаваме следното следствие.

Следствие 4.1. Нека $A_0 \overset{\ell}{\lhd} \omega$, където $\ell \geq 2$. Тогава съществуват променливи $A_1, \ldots, A_{\ell-1}$ и думи $\alpha_1, \ldots, \alpha_{\ell-1}, \beta, \gamma_1, \ldots, \gamma_{\ell-1}$, за които:

- $A_i \overset{\leq \ell-i}{\lhd} \alpha_{i+1} A_{i+1} \gamma_{i+1}$ за $i < \ell - 1$;
- $A_{\ell-1} \overset{1}{\lhd} \beta$;
- $\omega = \alpha_1 \cdots \alpha_{\ell-1} \beta \gamma_{\ell-1} \cdots \gamma_1$.

Упътване.

- Нека $A_0 \overset{\ell}{\lhd} \omega$. Щом имаме $\ell \geq 2$, то като приложим *Твърдение 4.1* получаваме, че можем да разбием думата ω на три части като $\omega = \alpha_1 \omega_1 \gamma_1$, за които съществува променлива A_1 с изводите $A_0 \overset{\leq \ell-0}{\lhd} \alpha_1 A_1 \gamma_1$ и $A_1 \overset{\ell-1}{\lhd} \omega_1$. Сега, ако $\ell - 1 = 1$, то полагаме $\beta = \omega_1$ и приключваме.
- В противен случай, щом $\ell - 1 \geq 2$, то отново прилагаме *Твърдение 4.1* и получаваме разбиване на ω_1 на три части като $\omega_1 = \alpha_2 \omega_2 \gamma_2$, за които съществува променлива A_1 с изводите и $A_1 \overset{\leq \ell-1}{\lhd} \alpha_2 A_2 \gamma_2$ и $A_2 \overset{\ell-2}{\lhd} \omega_2$. Обърнете внимание, че $\omega = \alpha_1 \alpha_2 \omega_2 \gamma_2 \gamma_1$. Отново, ако $\ell - 2 = 1$, то полагаме $\beta = \omega_2$ и приключваме.
- В противен случай, щом $\ell - 2 \geq 2$, продължаваме нататък и получаваме изводите $A_2 \overset{\leq \ell-2}{\lhd} \alpha_3 A_3 \gamma_3$ и $A_3 \overset{\ell-3}{\lhd} \omega_3$. Имаме, че $\omega = \alpha_1 \alpha_2 \alpha_3 \omega_3 \gamma_3 \gamma_2 \gamma_1$. Ако $\ell - 3 = 1$, то полагаме $\beta = \omega_3$ и приключваме.
- В крайна сметка, итерираме този процес $i = \ell - 1$ пъти.

□

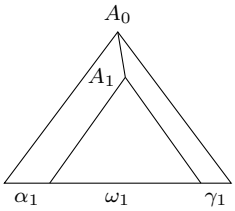
Ясно е от дефиницията, че ако $A \overset{\ell}{\lhd} \omega$, то съществува синтактично дърво с височина ℓ, корен A и листа ω. За граматиката G, нека

$$b \overset{\text{деф}}{=} \max\{|\alpha| \ : \ A \to \alpha \text{ е правило в } G\}.$$

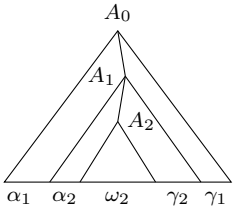
Числото b дава максималната разклоненност на синтактичните G-дървета. Знаем, връзката между броя на листата и височината на дървото: ако b-разклонено дърво има височина ℓ, то броят на листата е $\leq b^\ell$. Тогава, ако $A \overset{\ell}{\lhd} \omega$, то $|\omega| \leq b^\ell$. Оттук веднага получаваме и свойството, че ако $A \overset{m}{\lhd} \omega$ и $|\omega| > b^\ell$, то $m > \ell$.

Нека да обобщим горните разсъждения в следващото твърдение, на което ще се позовем по-късно.

(а) Синтактично дърво, което илюстрира изводите $A_0 \overset{\leq \ell}{\lhd} \alpha_1 A_1 \gamma_1$ и $A_1 \overset{\ell-1}{\lhd} \omega_1$:



(б) Синтактично дърво, което илюстрира изводите $A_1 \overset{\leq \ell-1}{\lhd} \alpha_2 A_2 \gamma_2$ и $A_2 \overset{\ell-2}{\lhd} \omega_2$:



Твърдение 4.2. Нека G е безконтекстна граматика и

$$b \stackrel{\text{деф}}{=} \max\{\, |\alpha| \,:\, A \to \alpha \text{ е правило в } G \,\}.$$

Тогава:

(1) Ако $A \overset{\ell}{\lhd} \omega$, то $|\omega| \leq b^\ell$.

(2) Ако $A \overset{m}{\lhd} \omega$ и $|\omega| > b^\ell$, то $m > \ell$.

Упътване. Можем да докажем (1) с индукция по ℓ следвайки дефиницията на релацията $\overset{\ell}{\lhd}$.

- $\ell = 0$. Тогава е ясно, че единственият случай за $A \overset{0}{\lhd} \omega$ е когато $\omega = A$ и тогава имаме $|\omega| \leq b^0 = 1$.
- Нека $\ell > 0$. Тук имаме два случая.
  - Нека $A \overset{1}{\lhd} \omega$ заради правилото $A \to \varepsilon$, т.е. $\omega = \varepsilon$. Тогава е ясно, че $|\varepsilon| = 0 \leq b^1$.
  - Нека $A \overset{\ell}{\lhd} \omega$ поради правилото

$$\frac{A \to_G X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \omega_1 \qquad \cdots \qquad X_n \overset{\ell_n}{\lhd} \omega_n}{A \overset{\ell}{\lhd} \underbrace{\omega_1 \cdots \omega_n}_{\omega}} \; (\ell = 1 + \sup\{\ell_1, \ldots, \ell_n\})$$

  От **(И.П.)** имаме, че $|\omega_i| \leq b^{\ell_i}$ за $i = 1, \ldots, n$. Поради избора на b имаме, че $n \leq b$. Тогава

$$|\omega| = \sum_{i=1}^{n} |\omega_i| \leq \sum_{i=1}^{n} b^{\ell_i} \leq b \cdot b^{\ell-1} = b^\ell.$$

□

Нека да разгледаме още две свойства на релацията за извод в граматика, които ще ни бъдат полезни по-късно.

Твърдение 4.3. За произволни променливи A, B и думи $\omega_1, \omega_2, \omega_3$ можем да направим извода:

$$\frac{A \overset{\ell}{\lhd} \omega_1 B \omega_3 \qquad B \overset{m}{\lhd} \omega_2}{A \overset{\leq \ell+m}{\lhd} \omega_1\omega_2\omega_3}$$

Упътване. Формално доказателството протича с индукция по ℓ.

- Нека $\ell = 0$. Тогава щом $A \overset{0}{\lhd} \omega_1 B \omega_3$ означава, че $\omega_1 = \omega_3 = \varepsilon$ и $A = B$. Тогава е ясно, че можем да заключим, че $A \overset{m}{\lhd} \omega_1\omega_2\omega_3$.
- Нека $\ell > 0$. От правило (2) следва, че

$$\frac{A \to X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \alpha_1 \qquad X_i \overset{\ell_i}{\lhd} \alpha_i' B \alpha_i'' \qquad X_n \overset{\ell_n}{\lhd} \alpha_n}{A \overset{\ell+1}{\lhd} \underbrace{\alpha_1 \cdots \alpha_i'}_{\omega_1} B \underbrace{\alpha_i'' \cdots \alpha_n}_{\omega_3}} \ (\ell = \sup\{\ell_1, \ldots, \ell_n\})$$

Понеже $\ell_i < \ell$, от можем да приложим **(И.П.)** , откъдето следва, че:

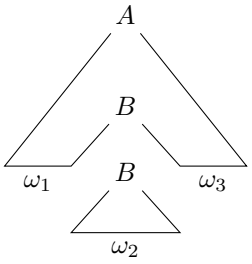
$$\frac{X_i \overset{\ell_i}{\lhd} \alpha_i' B \alpha_i'' \qquad B \overset{m}{\lhd} \omega_2}{X_i \overset{\leq \ell_i + m}{\lhd} \alpha_i' \omega_2 \alpha_i''}$$

Накрая пак прилагаме правило (2) за да получим следното:

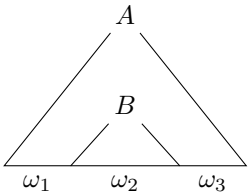
$$\frac{A \to X_1 \cdots X_n \qquad X_1 \overset{\ell_1}{\lhd} \alpha_1 \qquad X_i \overset{\leq \ell_i + m}{\lhd} \alpha_i' \omega_2 \alpha_i'' \qquad X_n \overset{\ell_n}{\lhd} \alpha_n}{A \overset{\leq \ell + m}{\lhd} \omega_1\omega_2\omega_3} \ (\ell = \sup\{\ell_1, \ldots, \ell_n\})$$

□

(а) Предпоставките на извода ни казват, че имаме синтактични дървета за изводите $A \overset{\ell}{\lhd} \omega_1 B \omega_3$ и $B \overset{m}{\lhd} \omega_2$



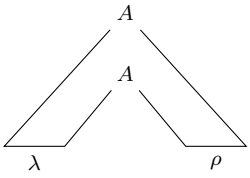
(б) Заключението на извода ни казва, че можем да съединим двете дървета и така да получим, че $A \overset{\leq \ell+m}{\lhd} \omega_1\omega_2\omega_3$



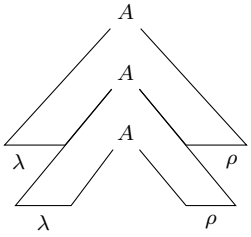
Твърдение 4.4. За произволна променлива A и произволни думи λ и ρ можем да направим извода:

$$\frac{A \overset{\ell}{\lhd} \lambda A \rho \qquad i \in \mathbb{N}}{A \overset{\leq i \cdot \ell}{\lhd} \lambda^i A \rho^i}$$

(а) Синтактично дърво за $A \overset{\ell}{\lhd} \lambda A \rho$



(б) Съединяваме две копия на дървото и получаваме синтактично дърво илюстриращо извода $A \overset{\leq 2\ell}{\lhd} \lambda^2 A \rho^2$



Доказателство. Индукция по i. Нека $i = 0$. Тогава директно имаме, че $A \overset{0}{\lhd} A$. Нека $i > 0$. Тогава получаваме извода:

$$\frac{A \overset{\ell}{\lhd} \lambda A \rho \qquad \dfrac{A \overset{\ell}{\lhd} \lambda A \rho \qquad A \overset{\leq (i-1)\cdot\ell}{\lhd} \lambda^{i-1} A \rho^{i-1}}{A \overset{\leq i \cdot \ell}{\lhd} \lambda^i A \rho^i}}{A \overset{\leq i \cdot \ell}{\lhd} \lambda^i A \rho^i} \ (\textit{Твърдение 4.3})$$

□

4.2 Еднозначни граматики (*)

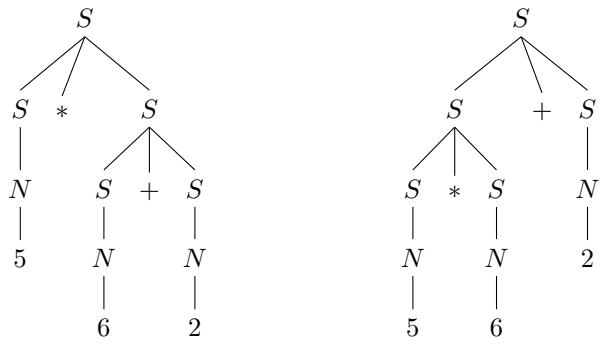
Практически синтактичните дървета са важни, когато искаме да зададем *смисъл* на една дума. Това е видно от *Пример 4.1*. Ако подходим наивно към задачата за построяване на граматика за аритметичните изрази, смисълът на аритметичните изрази може да бъде неясен.

Пример 4.1. Да разгледаме граматиката

$$G_0 = \begin{cases} S \to S + S \mid S * S \mid (S) \mid N \\ N \to 0 \mid 1 \mid 2 \mid \ldots \mid 9. \end{cases}$$

Подобен пример е разгледан и в [1]. Такъв вид граматики се наричат *неднозначни* (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 като и двете имат височина 4.



Фигура 4.7: Две G_0-дървета за извода $S \overset{4}{\lhd} 5 * 6 + 2$.

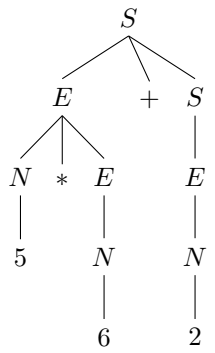
✍ Колко различни синтактични дървета за думата $1 + 5 * 6 + 2$ може да намерите?

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:

$$G_1 = \begin{cases} S \to E + S \mid E \\ E \to N \mid (S) \mid N * E \mid (S) * E \\ N \to 0 \mid 1 \mid 2 \mid \ldots \mid 9. \end{cases}$$

$*$ 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 4.8: Единственото G_1-дърво за извода $S \overset{4}{\lhd} 5 * 6 + 2$.

Пример 4.2. Да разгледаме граматиките:

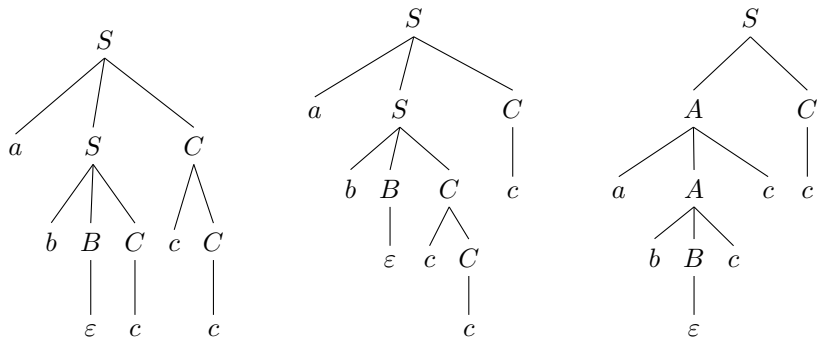
$$G_1 = \begin{cases} S \to aSC \mid bBC \mid \varepsilon \\ B \to bBC \mid \varepsilon \\ C \to cC \mid c; \end{cases} \qquad G_2 = \begin{cases} S \to AC \mid \varepsilon \\ A \to aAc \mid bBc \mid \varepsilon \\ B \to bBc \mid \varepsilon \\ C \to cC \mid \varepsilon. \end{cases}$$

Фигура 4.9: Две граматики за $L \overset{\text{деф}}{=} \{a^n b^m c^k \ : \ n + m \leq k\}$

✍ Съобразете, че е изпълнено:

$$L = \mathcal{L}(G_1) = \mathcal{L}(G_2).$$

✍ Съобразете, че всяка $\omega \in \mathcal{L}(G_2)$ има единствено синтактично G_2-дърво.
🔍Интересно е да се отбележи, че не съществува алгоритъм, който да отговаря на въпроса дали дадена безконтекстна граматика е еднозначна или не. Вижте *Следствие 6.9*.



(а) Първо G_1-дърво **(б)** Второ G_1-дърво **(в)** Единственото G_2-дърво

Фигура 4.10: Синтактични дървета за извода $S \lhd abccc$ относно граматиките G_1 и G_2.

4.3 Апроксимации на безконтекстен език

Обикновено използваме a, b, c за елементи на $\mathbf{\Sigma}$ и A, B, C за елементи на $\boldsymbol{V}$. Ще използваме X за елементи на $\mathbf{\Sigma} \cup \boldsymbol{V}$.

Да напомним, че когато говорихме за автомати, беше удобно да разсъждаваме за езика на всяка състояние $\mathcal{L}_{\mathcal{A}}(q)$. По подобен начин, сега ще дефинираме език $\mathcal{L}_G(X)$, за произволно $X \in \mathbf{\Sigma} \cup \boldsymbol{V}$ в граматиката G, по следния начин:

$$\mathcal{L}_G(X) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \,:\, X \lhd \omega\}.$$

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:

$$\mathcal{L}^\ell_G(X) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \,:\, X \stackrel{\leq \ell}{\lhd} \omega\}.$$

След един бърз поглед върху дефиницията на релацията $\lhd$ веднага съобразяваме, че ако $X \in \mathbf{\Sigma}$, то $\mathcal{L}^\ell_G(X) = \{X\}$ за всяко ℓ. Да видим сега какво става, когато $X \in \boldsymbol{V}$.

Задача 4.1. 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;
- $\mathcal{L}_G(A) = \bigcup_{\ell \geq 0} \mathcal{L}^\ell_G(A)$.

Нека тук отново да напомним, че за $n = 0$ имаме, че $X_1 \cdots X_n = \varepsilon$ и $\prod_{i=1}^n L_i = \{\varepsilon\}$. Това означава, че като частен случай получаваме $\mathcal{L}_G(\varepsilon) = \{\varepsilon\}$.

Ще се окаже удобно да обобщим езиците $\mathcal{L}_G(X)$ и $\mathcal{L}^\ell_G(X)$ за произволна дума $\alpha \in (\boldsymbol{V} \cup \mathbf{\Sigma})^\star$. Нека $\alpha = X_1 \cdots X_n$. Тогава

$$\mathcal{L}^\ell_G(X_1 \cdots X_n) = \mathcal{L}^\ell_G(X_1) \cdots \mathcal{L}^\ell_G(X_n);$$

$$\mathcal{L}_G(X_1 \cdots X_n) = \mathcal{L}_G(X_1) \cdots \mathcal{L}_G(X_n).$$

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

$$\bigcup\{\{1,2\},\{2,3\}\} = \{1,2\} \cup \{2,3\} = \{1,2,3\}.$$

Твърдение 4.5. Нека G е безконтекстна граматика и A е променлива в G. Тогава имаме следните зависимости:

$$\mathcal{L}_G^0(A) = \emptyset$$
$$\mathcal{L}_G^{\ell+1}(A) = \bigcup\{\mathcal{L}_G^\ell(\alpha) \,:\, A \to \alpha \text{ е правило в } G\}.$$

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 ω, за която $\omega \in \mathcal{L}_G^{\ell+1}(A)$, т.е. $A \overset{\leq \ell+1}{\lhd} \omega$. Ясно е, че е невъзможно да имаме $A \overset{0}{\lhd} \omega$. Тогава, според правилата на релацията $\lhd$, имаме следния извод:

Може да използваме и по-компактния запис

$$\mathcal{L}_G^{\ell+1}(A) = \bigcup_{A \to \alpha} \mathcal{L}_G^\ell(\alpha).$$

$$\frac{A \to_G X_1 \cdots X_n \qquad X_1 \overset{\leq \ell}{\lhd} \omega_1 \qquad \cdots \qquad X_n \overset{\leq \ell}{\lhd} \omega_n}{A \overset{\leq \ell+1}{\lhd} \underbrace{\omega_1 \cdots \omega_n}_{\omega}} \text{ правило (2)}$$

Понеже $X_i \overset{\leq \ell}{\lhd} \omega_i$, то по дефиниция имаме, че $\omega_i \in \mathcal{L}_G^\ell(X_i)$ за всяко $i = 1, 2, \ldots, n$. Тогава

$$\omega \in \bigcup\{\mathcal{L}_G^\ell(X_1) \cdots \mathcal{L}_G^\ell(X_n) \,:\, A \to_G X_1 \cdots X_n\}.$$

За да сме сигурни, че няма пробойни в нашите разсъждения, нека да разгледаме и частния случай на правило (2), което означихме като правило (3). Нека имаме извода $A \overset{1}{\lhd} \varepsilon$ получен чрез прилагане на правилото:

$$\frac{A \to_G \varepsilon}{A \overset{1}{\lhd} \varepsilon}$$

Понеже $\mathcal{L}_G^0(\varepsilon) = \{\varepsilon\}$, то е ясно, че $\varepsilon \in \bigcup\{\mathcal{L}_G^0(\alpha) \,:\, A \to \alpha \text{ е правило в } G\}$. Така доказахме включването ($\subseteq$).

Сега преминаваме към доказателството на включването ($\supseteq$). Нека вземем произволна дума ω принадлежаща на дясното множество. Това означава, че можем да представим думата като $\omega = \omega_1 \cdots \omega_n$, където $n \geq 1$ и $A \to_G X_1 \cdots X_n$ и $\omega_i \in \mathcal{L}_G^\ell(X_i)$. Получаваме следния извод:

$$\frac{A \to_G X_1 \cdots X_n \qquad \dfrac{\omega_1 \in \mathcal{L}_G^\ell(X_1)}{X_1 \overset{\leq \ell}{\lhd} \omega_1} \text{ (деф.)} \qquad \cdots \qquad \dfrac{\omega_n \in \mathcal{L}_G^\ell(X_n)}{X_n \overset{\leq \ell}{\lhd} \omega_n} \text{ (деф.)}}{A \overset{\leq \ell+1}{\lhd} \omega_1 \cdots \omega_n} \text{ правило (2)}$$

Заключаваме, че думата ω принадлежи на езика $\mathcal{L}^{\ell+1}(A)$.

В случая за $n = 0$, то получаваме, че $A \to_G \varepsilon$ и $\varepsilon \in \mathcal{L}_G^0(\varepsilon)$, откъдето по правило (3) получаваме извода $A \overset{1}{\lhd} \varepsilon$. Така доказахме и включването $(\supseteq)$. □

Понеже знаем, че $\mathcal{L}_G(A) = \bigcup_\ell \mathcal{L}_G^\ell(A)$ следващото следствие от *Твърдение 4.5* не е изненаващо.

Следствие 4.2. Нека G е безконтекстна граматика и A е променлива в G. Тогава

$$\mathcal{L}_G(A) = \bigcup\{\mathcal{L}_G(\alpha) \,:\, A \to \alpha \text{ е правило в } G\}.$$

Можем да използваме по-компактния запис

$$\mathcal{L}_G(A) = \bigcup_{A \to \alpha} \mathcal{L}_G(\alpha).$$

Доказателство. Нека $\omega \in \mathcal{L}_G(A)$, т.е. $\omega \in \mathcal{L}_G^{\ell+1}(A)$, за някое ℓ. Тогава, според *Твърдение 4.5*, $\omega \in \mathcal{L}_G^\ell(\alpha)$, за някое правило $A \to_G \alpha$. Оттук веднага получаваме, че $\omega \in \mathcal{L}_G(\alpha)$ и следователно

$$\omega \in \bigcup\{\mathcal{L}_G(\alpha) \,:\, A \to \alpha \text{ е правило в } G\}.$$

За обратната посока, нека $\omega \in \mathcal{L}_G(\alpha)$, където $A \to_G \alpha$ е правило в граматиката G. Това означава, че $\omega \in \mathcal{L}_G^\ell(\alpha)$, за някое ℓ, и следователно, според *Твърдение 4.5*, $\omega \in \mathcal{L}_G^{\ell+1}(A)$. Заключаваме, че $\omega \in \mathcal{L}_G(A)$. □

Пример 4.3. Да разгледаме безконтекстната граматика G зададена с правилата

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

От *Следствие 4.2* имаме следните равенства:

$$\begin{aligned}\mathcal{L}_G(S) &= \bigcup_{S \to \alpha} \mathcal{L}_G(\alpha) \\ &= \bigcup\{\mathcal{L}_G(aSb), \mathcal{L}_G(\varepsilon)\} \\ &= \{a\} \cdot \mathcal{L}_G(S) \cdot \{b\} \cup \{\varepsilon\}.\end{aligned}$$

С други думи, това означава, че езикът $\mathcal{L}_G(S)$ е решение на уравнението

$$X = \{a\} \cdot X \cdot \{b\} \cup \{\varepsilon\}.$$

4.4 Фундаментална теорема за безконтекстните езици

Да разгледаме произволна дума $\alpha \in (\mathbf{\Sigma} \cup \boldsymbol{V})^\star$. Понеже е възможно да имаме променливи в α, а променливите приемат стойности езици, то *стойността* на цялата дума α ще бъде език, който ще зависи от стойностите на променливите. И така, нека $\overrightarrow{K} = K_1, \ldots, K_n$ са произволни езици. Дефинираме езика, съответстващ на α, относно $\overrightarrow{K}$, с рекурсия:

$$
\begin{aligned}
&\mathcal{L}_{\overrightarrow{K}}(\varepsilon) \stackrel{\text{деф}}{=} \{\varepsilon\} \\
&\mathcal{L}_{\overrightarrow{K}}(\beta \cdot a) \stackrel{\text{деф}}{=} \mathcal{L}_{\overrightarrow{K}}(\beta) \cdot \{a\} && \text{// за } a \in \mathbf{\Sigma} \\
&\mathcal{L}_{\overrightarrow{K}}(\beta \cdot A_i) \stackrel{\text{деф}}{=} \mathcal{L}_{\overrightarrow{K}}(\beta) \cdot K_i && \text{// за } A_i \in \boldsymbol{V}.
\end{aligned}
$$

Ако имаме граматика G, то ние на практика знаем езиците, които съответстват на променливите, а именно, на променливата A съответства езика $\mathcal{L}_G(A)$. Да напомним, че вече дефинирахме стойността на α при граматиката G така:

$$
\begin{aligned}
&\mathcal{L}_G(\varepsilon) \stackrel{\text{деф}}{=} \{\varepsilon\} \\
&\mathcal{L}_G(\beta \cdot X) \stackrel{\text{деф}}{=} \mathcal{L}_G(\beta) \cdot \mathcal{L}_G(X).
\end{aligned}
$$

Понеже за $a \in \Sigma$, $\mathcal{L}_G(a) = \{a\}$, то за $K_1 = \mathcal{L}_G(A_1), \ldots, K_n = \mathcal{L}_G(A_n)$, получаваме, че $\mathcal{L}_{\overrightarrow{K}}(\alpha) = \mathcal{L}_G(\alpha)$.

Когато не знаем езиците $\overrightarrow{K} = K_1, \ldots, K_n$, можем поне да дефинираме с рекурсия теоретико-множествения израз $\mathcal{L}(\alpha)$, който съотетства на α така:

$$
\begin{aligned}
&\mathcal{L}(\varepsilon) \stackrel{\text{деф}}{=} \{\varepsilon\} \\
&\mathcal{L}(\beta \cdot a) \stackrel{\text{деф}}{=} \mathcal{L}(\beta) \cdot \{a\} && \text{// за } a \in \mathbf{\Sigma} \\
&\mathcal{L}(\beta \cdot A_i) \stackrel{\text{деф}}{=} \mathcal{L}(\beta) \cdot A_i && \text{// за } A_i \in \boldsymbol{V}.
\end{aligned}
$$

Тук в дясната страна на дефиницията променливата A_i играе ролята на неизвестен език.

Нека имаме безконтекстна граматика G, където $\boldsymbol{V} = \{A_1, \ldots, A_n\}$. Дефинираме система за G, където елементите на $\boldsymbol{V}$ играят ролята на неизвестни променливи, по следния начин:

$$
\left|\begin{array}{lcl}
A_1 &=& \bigcup\{\, \mathcal{L}(\alpha) \,:\, A_1 \to \alpha \text{ е правило в } G \,\} \\
&\vdots& \\
A_n &=& \bigcup\{\, \mathcal{L}(\alpha) \,:\, A_n \to \alpha \text{ е правило в } G \,\}.
\end{array}\right. \tag{4.1}
$$

Системата можем да запишем съкратено и така:

$$
\left|\begin{array}{lcl}
A_1 &=& \bigcup\limits_{A_1 \to \alpha} \mathcal{L}(\alpha) \\
&\vdots& \\
A_n &=& \bigcup\limits_{A_n \to \alpha} \mathcal{L}(\alpha).
\end{array}\right.
$$

Казваме, че $\overrightarrow{K} = K_1, \ldots, K_n$ е *решение* на системата (4.1), ако

$$
\left|\begin{array}{lcl}
K_1 &=& \bigcup\{\, \mathcal{L}_{\overrightarrow{K}}(\alpha) \,:\, A_1 \to \alpha \text{ е правило в } G \,\} \\
&\vdots& \\
K_n &=& \bigcup\{\, \mathcal{L}_{\overrightarrow{K}}(\alpha) \,:\, A_n \to \alpha \text{ е правило в } G \,\}.
\end{array}\right.
$$

Да напомним, че според *Следствие 4.2*, $\mathcal{L}_G(A_1), \ldots, \mathcal{L}_G(A_n)$ е решение на сис-

темата (4.1), което означава следното:

$$\left|\begin{array}{rcl} \mathcal{L}_G(A_1) & = & \bigcup\{\, \mathcal{L}_G(\alpha) \,:\, A_1 \to \alpha \text{ е правило в } G \,\} \\ & \vdots & \\ \mathcal{L}_G(A_n) & = & \bigcup\{\, \mathcal{L}_G(\alpha) \,:\, A_n \to \alpha \text{ е правило в } G \,\} \end{array}\right.$$

В този раздел ще видим каква е връзката между решенията на тази система и езиците $\mathcal{L}_G(A_1), \cdots \mathcal{L}_G(A_n)$.

Пример 4.4. Да разгледаме безконтекстната граматика

$$G = \begin{cases} A_1 \to aA_1c \mid A_2 \\ A_2 \to bA_2c \mid \varepsilon. \end{cases}$$

На нея съответства системата:

$$\left|\begin{array}{rcl} A_1 & = & \{a\}A_1\{c\} \cup A_2 \\ A_2 & = & \{b\}A_2\{c\} \cup \{\varepsilon\} \end{array}\right.$$

Двойката езици $\overrightarrow{K} \stackrel{\text{деф}}{=} K_1, K_2$ задават решение на системата, където

$$K_1 \stackrel{\text{деф}}{=} \{a^n b^k c^{n+k} \,:\, n, k \in \mathbb{N}\},$$
$$K_2 \stackrel{\text{деф}}{=} \{b^n c^n \,:\, n \in \mathbb{N}\}.$$

За да се уверим в това, трябва да проверим, че са изпълнени следните равенства:

$$K_1 = \mathcal{L}_{\overrightarrow{K}}(aA_1c) \cup \mathcal{L}_{\vec{K}}(A_2)$$
$$K_2 = \mathcal{L}_{\overrightarrow{K}}(bA_2c) \cup \mathcal{L}_{\vec{K}}(\varepsilon),$$

което се разписва така:

$$K_1 = \{a\}K_1\{c\} \cup K_2$$
$$K_2 = \{b\}K_2\{c\} \cup \{\varepsilon\}.$$

Непосредствено проверяваме, че

$$\{a^n b^k c^{n+k} \,:\, n, k \in \mathbb{N}\} = \{a\} \cdot \{a^n b^k c^{n+k} \,:\, n, k \in \mathbb{N}\} \cdot \{c\} \cup \{b^k c^k \,:\, k \in \mathbb{N}\},$$

откъдето следва, че $K_1 = \{a\}K_1\{c\} \cup K_2$, и

$$\{b^k c^k \,:\, k \in \mathbb{N}\} = \{b\} \cdot \{b^k c^k \,:\, k \in \mathbb{N}\} \cdot \{c\} \cup \{\varepsilon\},$$

откъдето следва, че $K_2 = \{b\}K_2\{c\} \cup \{\varepsilon\}$.

Интересен въпрос е дали тази граматика има други решения.

Лема 4.1. Нека имаме безконтекстна граматика G с променливи $V = \{A_1, \ldots, A_n\}$. Дефинираме система за G по следния начин:

$$\left|\begin{array}{rcl} A_1 & = & \bigcup\{\mathcal{L}(\alpha) \,:\, A_1 \to \alpha \text{ е правило в } G\} \\ & \vdots & \\ A_n & = & \bigcup\{\mathcal{L}(\alpha) \,:\, A_n \to \alpha \text{ е правило в } G\} \end{array}\right. \tag{4.2}$$

Тогава $\mathcal{L}_G(A_1), \ldots, \mathcal{L}_G(A_n)$ е *най-малкото решение* на системата.

Доказателство. Според *Следствие 4.2* вече знаем, че езиците $\mathcal{L}_G(A_i)$ са решения на системата. За да видим, че тези езици са *най-малкото решение*, ще използваме *Твърдение 4.5*, според което езиците $\mathcal{L}_G(A_i)$ могат да се представят чрез техните апроксимации. По-конкретно, ще използваме, че:

Имаме, че за всяко $i = 1, \ldots, n$,

$$\mathcal{L}_G(A_i) = \bigcup_{A_i \to \alpha} \mathcal{L}_G(\alpha).$$

$$\begin{aligned} \mathcal{L}^0_G(A_i) &= \emptyset \\ \mathcal{L}^{\ell+1}_G(A_i) &= \bigcup_{A_i \to \alpha} \mathcal{L}^{\ell}_G(\alpha). \end{aligned}$$

Нека езиците $K_1, \ldots, K_n$ задават произволно решение на системата. Понеже $\mathcal{L}_G(A_i) = \bigcup_\ell \mathcal{L}^\ell_G(A_i)$, достатъчно е да докажем, че $\mathcal{L}^\ell_G(A_i) \subseteq K_i$ за всяко ℓ.

Имаме, че за всяко $i = 1, \ldots, n$,

$$K_i = \bigcup_{A_i \to \alpha} \mathcal{L}_{\overrightarrow{K}}(\alpha).$$

- За $\ell = 0$ това е очевидно, защото $\mathcal{L}^0_G(A_i) = \emptyset$.
- Нека $\ell > 0$. Да разгледаме една дума $\omega \in \mathcal{L}^\ell_G(A_i)$. Да видим защо $\omega \in K_i$. Щом $\omega \in \mathcal{L}^\ell_G(A_i)$, то $\omega \in \mathcal{L}^{\ell-1}_G(\alpha)$ за някое правило $A_i \to \alpha$ в граматиката G. Достатъчно е да видим, че $\mathcal{L}^{\ell-1}_G(\alpha) \subseteq \mathcal{L}_{\overrightarrow{K}}(\alpha)$. Нека $\alpha = X_1 \ldots X_n$.
  - Ако $X_i \in \Sigma$, то $\mathcal{L}^{\ell-1}_G(X_i) = \{X_i\} = \mathcal{L}_{\overrightarrow{K}}(X_i)$.
  - Ако $X_i \in V$, то от **(И.П.)** следва, че $\mathcal{L}^{\ell-1}_G(X_i) \subseteq \mathcal{L}_{\overrightarrow{K}}(X_i)$.

  Оттук следва, че $\mathcal{L}^{\ell-1}_G(\alpha) \subseteq \mathcal{L}_{\overrightarrow{K}}(\alpha)$. Така заключаваме, че $\omega \in \mathcal{L}_{\overrightarrow{K}}(\alpha)$. Понеже $K_i = \bigcup_{A_i \to \alpha} \mathcal{L}_{\overrightarrow{K}}(\alpha)$, то $\omega \in K_i$.

□

Лема 4.2. Да фиксираме азбуката Σ. Да разгледаме следната произволна система от вида

$$\left|\begin{array}{rcl} A_1 & = & \bigcup\limits_{j=1}^{m_1} \mathcal{L}(\alpha_{1,j}) \\ & \vdots & \\ A_n & = & \bigcup\limits_{j=1}^{m_n} \mathcal{L}(\alpha_{n,j}), \end{array}\right. \tag{4.3}$$

където $\alpha_{i,j} \in (\{A_1, \ldots, A_n\} \cup \Sigma)^\star$ за всяко $i = 1, \ldots, n$ и $j = 1, \ldots, m_i$. Да дефинираме безконтекстната граматика G с променливи $V = \{A_1, \ldots, A_n\}$ и правила

$$A_i \to \alpha_{i,1} \mid \cdots \mid \alpha_{i,m_i},$$

за всяко $i = 1, \ldots, n$. Тогава $\mathcal{L}_G(A_1), \ldots, \mathcal{L}_G(A_n)$ е *най-малкото решение* на системата.

Доказателство. Нека имаме граматиката G от условието на лемата. От *Лема 4.1*, системата (4.2) съвпада точно със системата (4.3). Тогава според *Лема 4.1*, наистина $\mathcal{L}_G(A_1), \ldots, \mathcal{L}_G(A_n)$ е *най-малкото решение* на системата (4.3). □

Теорема 4.1 (Фундаментална теорема за безконтекстните езици)**.** Един език L е *безконтекстен* точно тогава, когато L е измежду езиците $L_1, \ldots, L_n$, които представляват *най-малкото решение* на система от вида

$$\left|\begin{array}{rcl} A_1 & = & \bigcup\limits_{j=1}^{m_1} \mathcal{L}(\alpha_{1,j}) \\ & \vdots & \\ A_n & = & \bigcup\limits_{j=1}^{m_n} \mathcal{L}(\alpha_{n,j}), \end{array}\right.$$

където $\alpha_{i,j} \in (\{A_1, \ldots, A_n\} \cup \Sigma)^\star$ за всяко $i = 1, \ldots, n$ и $j = 1, \ldots, m_i$.

Понеже системите, които описват регулярните езици са частен случай на системите, които описват безконтекстните езици, то веднага получаваме следното следствие.

Сравнете с *Теорема 3.5*.

Следствие 4.3. Всеки регулярен език е безконтекстен.

Доказателство. Нека имаме един регулярен език L. Тогава съществува детерминиран краен автомат $\mathcal{A}$ разпознаващ L и нека $s = q_1$. Според *Теорема 3.5*, съществува система от вида

$$\left|\begin{array}{rcl} X_1 & = & R_{1,1} \cdot X_1 \cup \cdots \cup R_{1,n} \cdot X_n \cup P_1 \\ & \vdots & \\ X_n & = & R_{n,1} \cdot X_1 \cup \cdots \cup R_{n,n} \cdot X_n \cup P_n, \end{array}\right. \tag{4.4}$$

Да напомним, че ако q_i е състояние на автомата $\mathcal{A}$, то асоциираме с него променливата X_i,

$$R_{i,j} \stackrel{\text{деф}}{=} \{x \in \Sigma : \boldsymbol{\delta}(q_i, x) = q_j\},$$

$P_i \stackrel{\text{деф}}{=} \{\varepsilon\}$, ако $q_i \in F$ и $P_i \stackrel{\text{деф}}{=} \emptyset$ иначе. Ако q_i е началното състояние на $\mathcal{A}$, то $L = \mathcal{L}_G(X_i)$.

където $P_i \subseteq \{\varepsilon\}$ и $R_{i,j} \subseteq \Sigma$, за всяко $i, j = 1, \ldots, n$, чието *единствено* решение е $L_1, \ldots L_n$ и $L_1 = L$.

Тази система представлява частен случай на системата в условието на *Теорема 4.1*. Това означава, че можем да приложим *Теорема 4.1* за системата (4.4). Така получаваме, че можем да построим безконтекстна граматика G с променливи $V = \{X_1, \ldots, X_n\}$ и $\mathcal{L}_G(X_1), \ldots, \mathcal{L}_G(X_n)$ е *най-малкото решение* на системата (4.4). Но понеже тази система има *единствено* решение, то нашият регулярен език $L = \mathcal{L}_G(X_1)$. Така заключаваме, че L е безконтекстен език. □

Нека да видим как можем да развием изложената техника за намиране на език по безконтекстна граматика.

Да напомним, че още в *Пример 2.3* видяхме, че езикът

$$L = \{a^n b^n \ : \ n \in \mathbb{N}\}$$

не е регулярен.

Пример 4.5. Да разгледаме безконтекстната граматика G зададена с правила:

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

Да разгледаме първите няколко члена на редицата от езици $\mathcal{L}^\ell_G(S)$:

$$\begin{aligned}
\mathcal{L}^0_G(S) &= \emptyset \\
\mathcal{L}^1_G(S) &= \mathcal{L}^0_G(aSb) \cup \mathcal{L}^0_G(\varepsilon) \\
&= \mathcal{L}^0_G(a) \cdot \mathcal{L}^0_G(S) \cdot \mathcal{L}^0_G(b) \cup \{\varepsilon\} \\
&= \{a\} \cdot \mathcal{L}^0_G(S) \cdot \{b\} \cup \{\varepsilon\} \\
&= \{a\} \cdot \emptyset \cdot \{b\} \cup \{\varepsilon\} \\
&= \{\varepsilon\} \\
\mathcal{L}^2_G(S) &= \{a\} \cdot \mathcal{L}^1_G(S) \cdot \{b\} \cup \{\varepsilon\} \\
&= \{a\} \cdot \{\varepsilon\} \cdot \{b\} \cup \{\varepsilon\} \\
&= \{\varepsilon, ab\} \\
\mathcal{L}^3_G(S) &= \{a\} \cdot \mathcal{L}^2_G(S) \cdot \{b\} \cup \{\varepsilon\} \\
&= \{a\} \cdot \{\varepsilon, ab\} \cdot \{b\} \cup \{\varepsilon\} \\
&= \{\varepsilon, ab, aabb\} = \{a^n b^n \ : \ n < 3\} \\
&\vdots
\end{aligned}$$

Лесно се доказва с индукция по ℓ, че $\mathcal{L}^\ell_G(S) = \{a^n b^n \ : \ n < \ell\}$. Заключаваме, че:

$$\mathcal{L}(G) = \mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}^\ell_G(S) = \{a^n b^n \ : \ n \in \mathbb{N}\}.$$

Сега да разгледаме граматиката G', където правилата са

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

Системата за тази граматика е

$$S = S \cup \{a\}S\{b\} \cup \{\varepsilon\}.$$

Всеки език, който разширява $\{a^n b^n \ : \ n \in \mathbb{N}\}$ е решение на системата.

Тук вече има смисъл да се говори за най-малко решение, защото е ясно, че езикът $\{a, b\}^\star$ също е решение на системата. Просто проверете, че е изпълнено равенството:

$$\{a, b\}^\star = \{a, b\}^\star \cup \{a\} \cdot \{a, b\}^\star \cdot \{b\} \cup \{\varepsilon\}.$$

Друго решение е $\{a\}^\star \cdot \{b\}^\star$. Отново да проверим:

$$\{a\}^\star \cdot \{b\}^\star = \{a\}^\star \cdot \{b\}^\star \cup \{a\} \cdot \{a\}^\star \cdot \{b\}^\star \cdot \{b\} \cup \{\varepsilon\}.$$

Най-малкото решение на тази система отново е езикът $\{a^n b^n \,:\, n \in \mathbb{N}\}$, защото непосредствено се проверява, че

$$\mathcal{L}_{G'}(S) = \{a^n b^n \,:\, n \in \mathbb{N}\}.$$

Примерни задачи

Задача 4.2. Докажете, че $L \stackrel{\text{деф}}{=} \{\omega\omega^{\text{rev}} : \omega \in \{a,b\}^\star\}$ е безконтекстен.

Упътване. Разгледайте граматиката

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

□

Задача 4.3. Докажете, че $L \stackrel{\text{деф}}{=} \{a^n b^k c^{n+k} : n,k \in \mathbb{N}\}$ 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 $\mathcal{L}(G) = \{a^n b^k c^{n+k} : n,k \in \mathbb{N}\}$. 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:

$$\mathcal{L}_G^\ell(S) \subseteq \{a^n b^k c^{n+k} : n,k \in \mathbb{N}\} \tag{4.5}$$
$$\mathcal{L}_G^\ell(B) \subseteq \{b^k c^k : k \in \mathbb{N}\}. \tag{4.6}$$

Това ще направим с индукция по ℓ. Да напомним, че според *Твърдение 4.5* имаме следните връзки:

$$\mathcal{L}_G^0(S) = \emptyset$$
$$\mathcal{L}_G^{\ell+1}(S) = \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\} \cup \mathcal{L}_G^\ell(B)$$
$$\mathcal{L}_G^0(B) = \emptyset$$
$$\mathcal{L}_G^{\ell+1}(B) = \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\} \cup \{\varepsilon\}.$$

Очевидно е, че *Свойство 4.5* и *Свойство 4.6* са изпълнени за $\ell = 0$. Да примем, че имаме *Свойство 4.5* и *Свойство 4.6* за някое ℓ. Обърнете внимание, че не можем да докажем *Свойство 4.5* независимо от *Свойство 4.6*. Ще докажем свойствата и за $\ell + 1$.

- Първо, нека $\alpha \in \mathcal{L}_G^{\ell+1}(S)$. Имаме два случая.
  - Нека $\alpha \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\}$. От **(И.П.)** следва, че $\alpha = a^{n+1} b^k c^{n+k+1}$ за някои естествени числа n и k. Тогава е ясно, че $\alpha \in \{a^n b^k c^{n+k} : n,k \in \mathbb{N}\}$.
  - Нека $\alpha \in \mathcal{L}_G^\ell(B)$. От **(И.П.)** следва, че $\alpha \in \{b^k c^k : k \in \mathbb{N}\} \subseteq \{a^n b^k c^{n+k} : n,k \in \mathbb{N}\}$.
- Второ, нека $\alpha \in \mathcal{L}_G^{\ell+1}(B)$. Имаме два случая.
  - Нека $\alpha \in \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\}$. От **(И.П.)** следва, че $\alpha = b^{k+1} c^{k+1}$ за някое естествено число k. Тогава е ясно, че $\alpha \in \{b^k c^k : k \in \mathbb{N}\}$.
  - Нека $\alpha = \varepsilon$. В този случай също е ясно, че

$$\alpha \in \{b^k c^k : k \in \mathbb{N}\}.$$

Оттук заключаваме, че $\mathcal{L}_G(S) \subseteq \{a^n b^k c^{n+k} : n,k \in \mathbb{N}\}$.

Сега да разгледаме *пълнотата* на граматиката, което означава, че всяка дума от езика се генерира от G. С други думи, $\{a^n b^k c^{n+k} : n,k \in \mathbb{N}\} \subseteq \mathcal{L}_G(S)$. Първо ще докажем, че

$$\{b^k c^k : k \in \mathbb{N}\} \subseteq \mathcal{L}_G(B). \tag{4.7}$$

Това ще направим с *пълна* индукция по дължината на думата. Да разгледаме произволна дума $\alpha \in \{b^k c^k : k \in \mathbb{N}\}$.

- 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 4.7* имаме $\alpha \in \{b\} \cdot \mathcal{L}_G(B) \cdot \{c\} \subseteq \mathcal{L}_G(B)$.

Така доказахме *Свойство 4.7*. Сега ще докажем, че:

$$\{a^n b^k c^{n+k} : n,k \in \mathbb{N}\} \subseteq \mathcal{L}_G(S). \tag{4.8}$$

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 4.7* следва, че $\alpha \in \mathcal{L}_G(B) \subseteq \mathcal{L}_G(S)$.

– Нека $n > 0$. Тогава от **(И.П.)** за *Свойство 4.8* 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

$$\mathcal{L}(G) = \{a^n b^k c^{n+k} \ : \ n, k \in \mathbb{N}\}.$$

□

Задача 4.4. Докажете, че $L \overset{\text{деф}}{=} \{a^m b^n c^k \ : \ m + n \geq k\}$ е безконтекстен.

Доказателство. Да разгледаме граматиката

$$G = \begin{cases} S \to aSc \mid aS \mid B \\ B \to bBc \mid bB \mid \varepsilon. \end{cases}$$

От *Твърдение 4.5* имаме, че:

$$\begin{aligned} \mathcal{L}_G^0(S) &= \emptyset \\ \mathcal{L}_G^{\ell+1}(S) &= \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\} \cup \{a\} \cdot \mathcal{L}_G^\ell(S) \cup \mathcal{L}_G^\ell(B) \\ \mathcal{L}_G^0(B) &= \emptyset \\ \mathcal{L}_G^{\ell+1}(B) &= \{b\} \cdot \mathcal{L}_G^\ell(B) \cdot \{c\} \cup \{b\} \cdot \mathcal{L}_G^\ell(B) \cup \{\varepsilon\}. \end{aligned}$$

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 $\ell = 0$.

$$\mathcal{L}_G^\ell(S) \subseteq \{a^n b^m c^k \ : \ n + m \geq k\} \tag{4.9}$$
$$\mathcal{L}_G^\ell(B) \subseteq \{b^m c^k \ : \ m \geq k\}. \tag{4.10}$$

Ще докажем, че:

$$\begin{aligned} \mathcal{L}_G^{\ell+1}(S) &\subseteq \{a^n b^m c^k \ : \ n + m \geq k\} \\ \mathcal{L}_G^{\ell+1}(B) &\subseteq \{b^m c^k \ : \ m \geq k\}. \end{aligned}$$

За първото включване, да разгледаме произволна дума $\alpha \in \mathcal{L}_G^{\ell+1}(S)$. Имаме три случая:

- Ако $\alpha \in \mathcal{L}_G^\ell(B)$, то от **(И.П.)** имаме, че

$$\alpha \in \{b^m c^k \ : \ m \geq k\} \subseteq \{a^n b^m c^k \ : \ n + m \geq k\}.$$

- Ако $\alpha \in \{a\} \cdot \mathcal{L}_G^\ell(S)$, то от **(И.П.)** имаме, че

$$\alpha \in \{a^{n+1} b^m c^k \ : \ n+m \geq k\} \subseteq \{a^n b^m c^k \ : \ n+m \geq k\}.$$

- Ако $\alpha \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{c\}$, то от **(И.П.)** имаме, че

$$\alpha \in \{a^{n+1} b^m c^{k+1} \ : \ n+m \geq k\} \subseteq \{a^n b^m c^k \ : \ n+m \geq k\}.$$

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:

$$\alpha \in \{b^{m+1} c^{k+1} \ : \ m \geq k\} \subseteq \{b^m c^k \ : \ m \geq k\}.$$

- Нека $\alpha \in \{b\} \cdot \mathcal{L}_G^\ell(B)$. Тогава от **(И.П.)** имаме, че:

$$\alpha \in \{b^{m+1} c^k \ : \ m \geq k\} \subseteq \{b^m c^k \ : \ m \geq k\}.$$

- Нека $\alpha \in \{\varepsilon\}$. Тогава е ясно, че имаме $\alpha \in \{b^m c^k \ : \ m \geq k\}$.

Така доказахме *коректност* на граматиката. Най-накрая заключаваме, че

$$\begin{aligned} \mathcal{L}_G(S) &= \bigcup_\ell \mathcal{L}_G^\ell(S) \subseteq \{a^n b^m c^k \ : \ n + m \geq k\} \\ \mathcal{L}_G(B) &= \bigcup_\ell \mathcal{L}_G^\ell(B) \subseteq \{a^n b^m c^k \ : \ n + m \geq k\}. \end{aligned}$$

Сега ще докажем *пълнота* на граматиката. Тук ще използваме представянията на $\mathcal{L}_G(S)$ и $\mathcal{L}_G(B)$ според *Следствие 4.2*. Сега трябва да докажем обратните включвания, а именно:

$$\{a^n b^m c^k \ : \ n + m \geq k\} \subseteq \mathcal{L}_G(S) \tag{4.11}$$
$$\{b^m c^k \ : \ m \geq k\} \subseteq \mathcal{L}_G(B). \tag{4.12}$$

Трябва да започнем първо със *Свойство 4.12*. 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 4.11*. 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 4.12* 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 4.5. Докажете, че езикът

$$L = \{a^n b^m c^k d^\ell \ : \ n + k = m + \ell\}$$

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:

$$L_1 = \{a^n b^m c^k \ : \ m = n + k\},$$
$$L_2 = \{b^m c^k d^\ell \ : \ k = m + \ell\}.$$

Така получаваме, че

$$L = \{a^n \cdot \omega \cdot d^n \ : \ n \in \mathbb{N} \,\&\, \omega \in L_1 \cup L_2\}.$$

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 4.6. Докажете, че езикът

$$L = \{\alpha\beta \in \{a, b\}^\star \ : \ |\alpha| = |\beta| \,\&\, \alpha \neq \beta\}$$

е безконтекстен.

Упътване. Да разгледаме една произволна дума ω, където $\omega = \alpha\beta$, $|\alpha| = |\beta|$ и $\alpha \neq \beta$. Знаем, че съществува индекс $i < |\alpha|$, такъв че думата ω може да се представи така:

$$\omega = \alpha[:i] \cdot \alpha[i] \cdot \alpha[i+1:] \cdot \beta[:i] \cdot \beta[i] \cdot \beta[i+1:],$$

където $\alpha[i] \neq \beta[i]$.

Нека $n = |\alpha| = |\beta|$ и да представим n като $n = i + 1 + k$. Имаме два случая. Първо, ако $k \geq i$, то можем да преставим ω по следния начин:

$$\underbrace{\alpha[:i]}_{\text{дълж. } i} \cdot \alpha[i] \cdot \underbrace{\alpha[i+1:2i+1]}_{\text{дълж. } i} \cdot \underbrace{\alpha[2i+1:] \cdot \beta[:i]}_{\text{дълж. } k} \cdot \beta[i] \cdot \underbrace{\beta[i+1:]}_{\text{дълж. } k}.$$

И второ, ако $k < i$, то можем да преставим ω по следния начин:

$$\underbrace{\alpha[:i]}_{\text{дълж. } i} \cdot \alpha[i] \cdot \underbrace{\alpha[i+1:] \cdot \beta[:i-k]}_{\text{дълж. } i} \cdot \underbrace{\beta[i-k:i]}_{\text{дълж. } k} \cdot \beta[i] \cdot \underbrace{\beta[i+1:]}_{\text{дълж. } k}.$$

От тези представяния на ω е ясно, че можем да изразим езика L като $L = L_a \cdot L_b \cup L_b \cdot L_a$, където:

$$L_a = \{\alpha \cdot a \cdot \beta \ : \ \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta|\}$$
$$L_b = \{\alpha \cdot b \cdot \beta \ : \ \alpha, \beta \in \{a, b\}^\star \,\&\, |\alpha| = |\beta|\}.$$

Сега разгледайте безконтекстната граматика

$$G = \begin{cases} S \to AB \mid BA \\ A \to XAX \mid a \\ B \to XBX \mid b \\ X \to a \mid b. \end{cases}$$

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 4.7. Докажете, че езикът

$$L = \{\alpha\sharp\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, \alpha \neq \beta\}$$

е безконтекстен.

Упътване. Разгледайте граматиката:

$$G = \begin{cases} S \to AaR \mid BbR \mid E \\ A \to XAX \mid bR\sharp \\ B \to XBX \mid aR\sharp \\ E \to XEX \mid XR\sharp \mid \sharp XR \\ R \to XR \mid \varepsilon \\ X \to a \mid b. \end{cases}$$

Имаме, че за произволни думи $\alpha, \beta, \gamma, \delta \in \{a,b\}^\star$,

$$S \overset{\star}{\lhd} \alpha b\gamma\sharp\beta a\delta \,\&\, |\alpha| = |\beta|,$$
$$S \overset{\star}{\lhd} \alpha a\gamma\sharp\beta b\delta \,\&\, |\alpha| = |\beta|, \text{ или}$$
$$S \overset{\star}{\lhd} \alpha\sharp\beta \,\&\, |\alpha| \neq |\beta|.$$

□

Задача 4.8. Докажете, че езикът

$$L = \{a^n b^m c^k d^\ell \,:\, n + k \geq m + \ell\}$$

е безконтекстен.

Задача 4.9. Докажете, че езикът

$$L = \{\alpha\sharp\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, |\alpha| \neq |\beta|\}$$

е безконтекстен.

4.5 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.

[22, стр. 125], [10, стр. 125], [9, стр. 275], [13, стр. 148].

Лема 4.3 (за покачването (безконтекстни езици)). За всеки безконтекстен език L съществува число $p > 0$, 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) $(\forall n \in \mathbb{N})[xy^n uv^n w \in L]$.

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. Без ограничение на общността, нека G да бъде в нормална форма на Чомски. Тогава максималната разклоненост на всяко синтактично G-дърво е 2. Да положим

$$p \stackrel{\text{деф}}{=} 2^{|\boldsymbol{V}|+1}.$$

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 $S \lhd \alpha$, според *Твърдение 4.2*, винаги имаме $S \overset{\ell}{\lhd} \alpha$ за $\ell \geq |\boldsymbol{V}| + 1$. Тогава според *Следствие 4.1* получаваме редицата от променливи $A_0, A_1, \ldots, A_{\ell-1}$ и думи $\lambda_1, \ldots, \lambda_{\ell-1}, \mu, \rho_{\ell-1}, \ldots, \rho_1$, където:

Важното е да вземем дума $\alpha \in \mathcal{L}(G)$, за която $|\alpha| > 2^{|\boldsymbol{V}|}$.

- $A_0 = S$,
- $A_i \overset{\ell-i}{\lhd} \lambda_{i+1}A_{i+1}\rho_{i+1}$ за $i < \ell - 1$,
- $A_{\ell-1} \overset{1}{\lhd} \mu$,
- $\alpha = \lambda_1 \cdots \lambda_{\ell-1}\mu\rho_{\ell-1} \cdots \rho_1$.

Да разгледаме множеството от двойки индекси

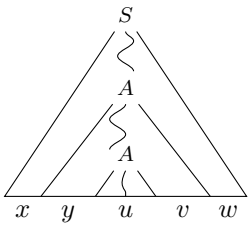
$$P \stackrel{\text{деф}}{=} \{\langle i, j\rangle \,:\, i < j < \ell \,\&\, A_i = A_j\}.$$

Понеже $\ell \geq |\boldsymbol{V}| + 1$, то от принципа на Дирихле, множеството P е непразно. Да изберем лексикографски най-голямата двойка $\langle i, j\rangle \in P$ и да разгледаме

изводите:

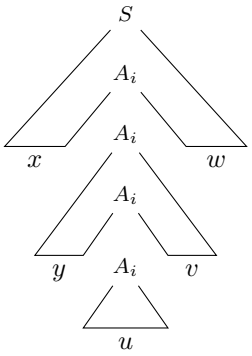
$$A_0 \lhd \underbrace{\lambda_1 \cdots \lambda_i}_{x} A_i \underbrace{\rho_i \cdots \rho_1}_{w},$$
$$A_i \lhd \underbrace{\lambda_{i+1} \cdots \lambda_j}_{y} A_j \underbrace{\rho_j \cdots \rho_{i+1}}_{v},$$
$$A_j \lhd \underbrace{\lambda_{j+1} \cdots \lambda_{\ell-1} \mu \rho_{\ell-1} \cdots \rho_{j+1}}_{u}.$$

(а) Разбиваме дървото на три части.



Сега остава да проверим условието на лемата:

(б) Получаваме три отделни синтактични дървета.



- Възможно ли е $A_i \overset{>|\boldsymbol{V}|+1}{\lhd} yuv$? Ако това е така, то от принципа на Дирихле ще имаме двойка $\langle i', j' \rangle \in P$, която е лексикографски по-голяма от $\langle i, j \rangle$. Това би било противоречие. Следователно $A_i \overset{\leq|\boldsymbol{V}|+1}{\lhd} yuv$, откъдето заключаваме, че $|yuv| \leq p = 2^{|\boldsymbol{V}|+1}$.
- Понеже граматиката G е в нормална форма на Чомски, веднага получаваме, че $|yv| \geq 1$.
- Понеже $A_i = A_j$, то за произволно $n \in \mathbb{N}$ имаме извода:

$$\dfrac{\dfrac{\dfrac{A_i \lhd yA_iv}{A_i \lhd y^nA_iv^n}\,(\textit{Тв. 4.4}) \qquad A_i \lhd u}{A_i \lhd y^nuv^n}\,(\textit{Тв. 4.3}) \qquad S \lhd xA_iw}{S \lhd xy^nuv^nw}\,(\textit{Тв. 4.3})$$

Оттук заключаваме, че $(\forall n \in \mathbb{N})[xy^nuv^nw \in \mathcal{L}(G)]$.

□

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 4.4. Нека L е произволен **безкраен** език. Нека също така е изпълнено, че:

Ако L е краен език, то е ясно, че L е безконтекстен.

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

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

($\forall$) за *всяко* разбиване на думата на пет части, $\alpha = xyuvw$, със свойствата $|yv| \geq 1$ и $|yuv| \leq p$,

($\exists$) 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 3.4*

✍ Докажете!

Следствие 4.5. 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 $p \leq |\alpha| < 2p$.

✍ Защо?

Забележка. Алгоритъм за проверка дали един безконтекстен език е безкраен следвайки горния критерий би имал експоненциална сложност относно $|G|$.

Пример 4.6. Да видим защо езикът $L = \{a^n b^n c^n \ : \ n \in \mathbb{N}\}$ не е безконтекстен.

$(\forall)$ Разглеждаме произволна константа $p \geq 1$.

$(\exists)$ Избираме дума $\alpha \in L$ със свойството, че $|\alpha| \geq p$. В случая, нека $\alpha = a^p b^p c^p$.

$(\forall)$ Разглеждаме произволно разбиване на α на пет части $\alpha = xyuvw$ със свойствата $|yuv| \leq p$ и $1 \leq |yv|$.

✍ Съобразете, че може да изберете и $i = 0$. В този пример може да вземете едно и също i за всяко разбиване на думата α от предишната стъпка. В други примери ще видим, че изборът на i зависи от разглежданото разбиване на думата.

$(\exists)$ За всяко такова разбиване ще посочим i, за което $xy^i uv^i w \notin L$. Знаем, че поне едно от y и v не е празната дума. Имаме няколко случая за y и v.

- y и v са думи съставени от една буква. В този случай получаваме, че xy^2uv^2w има различен брой букви a, b и c.
- y или v е съставена от две букви. В този случай получаваме, че в xy^2uv^2w редът на буквите е нарушен.
- понеже $|yuv| \leq p$, то не е възможно в y или v да се срещат и трите букви.

Оказа се, че във всички възможни случаи за y и v, $xy^2uv^2w \notin L$.

Така от *Следствие 4.4* следва, че езикът L не е безконтекстен.

Незатвореност относно булевите операции

Твърдение 4.6. Безконтекстните езици **не** са затворени относно операциите сечение и допълнение.

Упътване. Да разгледаме езика $L_0 = \{a^n b^n c^n \ : \ n \in \mathbb{N}\}$, за който вече знаем от *Пример 4.6*, че не е безконтекстен. Да вземем също така и безконтекстните езици

✍ Защо L_1 и L_2 са безконтекстни?

$$L_1 = \{a^n b^n c^m \ : \ n, m \in \mathbb{N}\},$$
$$L_2 = \{a^m b^n c^n \ : \ n, m \in \mathbb{N}\}.$$

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

  $$L_0 = L_1 \cap L_2 = \overline{\overline{L}_1 \cup \overline{L}_2} = \overline{L}_3$$

  е безконтекстен, което е противоречие.

Друг пример, с който може да се види, че безконтекстните езици не са затворени относно допълнение е като се докаже, че езикът

$$\{a, b\}^\star \setminus \{\omega\omega \ : \ \omega \in \{a, b\}^\star\}$$

е безконтекстен. Това следва лесно като се използва *Задача 4.6*.

□

Примерни задачи

Задача 4.10. Докажете, че езикът

$$L = \{a^i b^j c^k \ : \ 0 \leq i \leq j \leq k\}$$

не е безконтекстен.

Упътване. Прилагаме *Следствие 4.4*.

($\forall$) 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 4.11. Докажете, че езикът

$$L = \{ \ \alpha \cdot \alpha \ : \ \alpha \in \{a,b\}^\star \ \}$$

не е безконтекстен.

Упътване. Разгледайте $\omega = a^p b^p a^p b^p$. □

Задача 4.12. Докажете, че езикът

$$L = \{\alpha\beta\alpha^{\mathtt{rev}} \ : \ \alpha, \beta \in \{a,b\}^\star \ \& \ |\alpha| = |\beta|\}$$

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 4.13. Докажете, че езикът

$$L = \{\alpha\beta\alpha \ : \ \alpha, \beta \in \{a,b\}^\star \ \& \ |\alpha| \geq 1\}$$

не е безконтекстен.

Упътване.

- Защо не става с $\omega = a^p b a^p b$?
- Защо не става с $\omega = ab^p ab^p$?
- Пробвайте с $\omega = a^p b^p a^p b^p$.

□

Задача 4.14. Докажете, че езикът

$$L = \{\alpha\sharp\beta \ : \ \alpha \text{ е подниз на } \beta\}$$

не е безконтекстен.

Упътване.

- Защо не става ако вземем $\omega = a^p \sharp a^p$?
- Защо не става ако вземем $\omega = a^p b \sharp a^p b$?
- Разгледайте $\omega = a^p b^p \sharp a^p b^p$.

□

Задача 4.15. Вярно ли е, че следните езици са безконтекстни:

а) $L = \{\alpha\sharp\beta \ : \ \alpha, \beta \in \{0,1\}^\star \ \& \ \overline{\alpha}_{(2)} + 1 = \overline{\beta}_{(2)}\}$;

б) $L = \{\alpha\sharp\beta^{\mathtt{rev}} \ : \ \alpha, \beta \in \{0,1\}^\star \ \& \ \overline{\alpha}_{(2)} + 1 = \overline{\beta}_{(2)}\}$?

4.6 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

$$|G| \stackrel{\text{деф}}{=} |V| + |\Sigma| + \sum_{(A,\alpha)\in R} |A\alpha|.$$

4.7 Опростяване на безконтекстни граматики

Премахване на безполезните променливи

[10, стр. 88], [9, стр. 256].

Нека е дадена безконтекстната граматика G. Една променлива A се нарича **полезна**, ако съществува извод от следния вид:

$$S \lhd_G \lambda A \rho \lhd_G \omega,$$

където $\omega \in \mathbf{\Sigma}^\star$, а $\lambda, \rho \in (\boldsymbol{V} \cup \mathbf{\Sigma})^\star$. 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 4.4. 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 $A' \in \boldsymbol{V}'$, $\mathcal{L}_{G'}(A') \neq \emptyset$.

Упътване. Целта ни е да намерим множеството `Gen` от променливи, които генерират думи, т.е. търсим множеството

$$\texttt{Gen} \stackrel{\text{деф}}{=} \{A \in \boldsymbol{V} \,:\, \mathcal{L}_G(A) \neq \emptyset\}.$$

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]` $\stackrel{\text{деф}}{=} \emptyset$;
- `Gen[n+1]` $\stackrel{\text{деф}}{=} \{A \in \boldsymbol{V} \,:\, (\exists \omega \in (\mathbf{\Sigma} \cup$ `Gen[n]` $)^\star)[A \to_G \omega]\}$.
- 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:

$$A \in \texttt{Gen[k]} \;\Leftrightarrow\; (\exists \omega \in \mathbf{\Sigma}^\star)[A \stackrel{\leq k}{\lhd}_G \omega],$$

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 4.6. Съществува *полиномиален* алгоритъм, който определя за всяка безконтекстна граматика G дали $\mathcal{L}(G) = \emptyset$.

Доказателство. Прилагаме алгоритъма от *Лема 4.4* и намираме множеството от променливи $\boldsymbol{V}'$. Тогава $\mathcal{L}(G) = \emptyset$ точно тогава, когато $S \notin \boldsymbol{V}'$. □

Лема 4.5. Съществува алгоритъм, който по дадена безконтекстна граматика $G = \langle \boldsymbol{V}, \boldsymbol{\Sigma}, \boldsymbol{R}, \boldsymbol{S} \rangle$, намира $G' = \langle \boldsymbol{V}', \boldsymbol{\Sigma}, \boldsymbol{R}', \boldsymbol{S} \rangle$, $\mathcal{L}(G') = \mathcal{L}(G)$, със свойството, че за всяко $B \in \boldsymbol{V}'$ съществуват λ и ρ, за които $S \lhd_{G'} \lambda B \rho$, т.е. всяка променлива в G' е достижима от началната променлива $\boldsymbol{S}$.

Упътване. Нашата цел е да намерим множеството

$$\texttt{Reach} \stackrel{\text{деф}}{=} \{B \in \boldsymbol{V} \,:\, \boldsymbol{S} \lhd \lambda B \rho \text{ за някои } \lambda \text{ и } \rho\}.$$

Новата граматика G' ще има променливи $V' \stackrel{\text{деф}}{=} \texttt{Reach}$, като правилата на граматика G' са същите като тези на G, но ограничени до $\boldsymbol{V}'$. 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 $\texttt{Reach[i]} \subseteq \boldsymbol{V}$ по следния начин:

$$\begin{aligned}
\texttt{Reach[0]} &\stackrel{\text{деф}}{=} \{\boldsymbol{S}\} \\
\texttt{Reach[i+1]} &\stackrel{\text{деф}}{=} \{B \in \boldsymbol{V} \,:\, (\exists A \in \texttt{Reach[i]})(\exists \lambda)(\exists \rho)[\, A \to_G \lambda B \rho \,]\} \\
&\quad \cup \texttt{Reach[i]}.
\end{aligned}$$

Докажете, че за всяко i е изпълнено:

$$\texttt{Reach[i]} = \{B \in \boldsymbol{V} \,:\, \boldsymbol{S} \stackrel{\leq i}{\lhd} \lambda B \rho \text{ за някои } \lambda \text{ и } \rho\}.$$

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.

$$\texttt{Reach[n]} = \{B \in \boldsymbol{V} \,:\, \boldsymbol{S} \lhd \lambda B \rho \text{ за някои } \lambda \text{ и } \rho\}.$$

□

Теорема 4.2. 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.

Упътване. Прилагаме върху G първо процедурата от *Лема 4.4* и след това върху резултата прилагаме процедурата от *Лема 4.5*. □

Забележка. Защо е важна последователността на прилагане? Например, да разгледаме граматиката

$$G = \begin{cases} S \to AB \mid ab \\ A \to aA \mid \varepsilon \\ B \to bB. \end{cases}$$

Ако първо приложим процедурата от *Лема 4.5*, то нищо няма да премахнем, защото A и B са достижими от S. След това, ако приложим процедурата от *Лема 4.4*, ще премахнем всички правила, в които участва променливата B, т.е. получаваме граматика G' с правила $S \to ab$ и $A \to aA \mid \varepsilon$. Така A става недостижима променлива от S и трябва пак да приложим процедурата от *Лема 4.5*. Алгоритъмът описан тук е с квадратична сложност. Има и линеен такъв. Вижте [9, стр. 296].

Премахване на ε-правила

Лема 4.6. 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

$$\texttt{E} = \{A \in \boldsymbol{V} \ : \ A \lhd \varepsilon\}.$$

За да направим това, следваме процедурата:

На всяка стъпка трябва да обходим цялата граматика, следователно всяка итерация отнема $\mathcal{O}(|G|)$ време.

1) Първо строим множествата `E[n]` по следния начин:

   - `E[0]` $\stackrel{\text{деф}}{=} \emptyset$;

- $\texttt{E[n+1]} \stackrel{\text{деф}}{=} \{B \in \boldsymbol{V} \,:\, (\exists \omega \in \texttt{E[n]}^\star)[B \to \omega]\}$.

Обърнете внимание, че имаме $\texttt{E[n]}^\star$, а не просто $\texttt{E[n]}$. Също така да напомним, че $\emptyset^\star = \{\varepsilon\}$.

- Докажете, че за всяко n е изпълнено, че

$$\texttt{E[n]} = \{A \in \boldsymbol{V} \,:\, A \stackrel{\leq n}{\lhd} \varepsilon\}.$$

- Спираме на първото n, за което $\texttt{E[n]} = \texttt{E[n+1]}$. Лесно се съобразява, че за това n е изпълнено, че

Намираме множеството E за време $\mathcal{O}(|G|^2)$.

$$\texttt{E[n]} = \{A \in \boldsymbol{V} \,:\, A \lhd \varepsilon\}.$$

2) Дефинираме новата граматика G' така, че G' съдържа правило $A \to X_0 \cdots X_n$ точно тогава, когато $A \to \omega_0 X_0 \omega_1 X_1 \cdots X_n \omega_{n+1}$ е правило в G за някои думи $\omega_i \in \texttt{E}^\star$. Броят на правилата може да се увеличи *експоненциално*!. Това е така, защото в най-лошия случай извеждаме всички подмножества на дадено множество от променливи.

   Лесно се съобразява, че $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

✍ Докажете сами!

□

Премахване на преименуващи правила

Едно правило в граматиката G се нарича **преименуващо**, ако е от вида $A \to_G B$. Нека е дадена граматика $G = \langle \boldsymbol{V}, \boldsymbol{\Sigma}, \boldsymbol{R}, \boldsymbol{S} \rangle$. 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 $A \lhd B$, ако $B \to \alpha$ е правило в граматиката G, което не е преименуващо, то добавяме към G' правилото $A \to \alpha$.

В [9, стр. 263] преименуващите правила се наричат *unit productions*.

Лема 4.7. 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:

$$\texttt{Unit} = \{(A, B) \in \boldsymbol{V} \times \boldsymbol{V} \,:\, A \stackrel{\star}{\lhd} B\}.$$

Отново, за да намерим Unit, ние строим неговите апроксимации Unit[n], където:

$$\texttt{Unit[0]} \stackrel{\text{деф}}{=} \{(A, A) \,:\, A \in \boldsymbol{V}\}$$

$$\texttt{Unit[n+1]} \stackrel{\text{деф}}{=} \texttt{Unit[n]} \cup \{(A, B) \,:\, (\exists C)[A \to_G C \;\&\; (C, B) \in \texttt{Unit[n]}]\}.$$

Лесно се съобразява, че имаме свойството:

$$\texttt{Unit[n]} = \{(A,B) \in \boldsymbol{V} \times \boldsymbol{V} \,:\, A \overset{\leq n}{\lhd} B\}.$$

`Unit` има големина $\mathcal{O}(|G|^2)$.

Спираме на първото n, за което `Unit[n]` $=$ `Unit[n+1]`. Тогава `Unit` $=$ `Unit[n]`. Нека $R'_0 \overset{\text{деф}}{=} R \setminus (\boldsymbol{V} \times \boldsymbol{V})$ са правилата на R, които не са преименуващи. Тогава правилата на новата граматика G' ще бъдат:

$$R' \overset{\text{деф}}{=} \{\ (A,\alpha) \in \boldsymbol{V} \times (\boldsymbol{V} \cup \boldsymbol{\Sigma})^\star \,:\, (\exists B)[(A,B) \in \texttt{Unit}\ \&\ (B,\alpha) \in R'_0]\ \}.$$

Обърнете внимание, че понеже $(A,A) \in$ `Unit` за всяко $A \in \boldsymbol{V}$, то $R'_0 \subseteq R'$. Лесно се съобразява, че $\mathcal{L}(G) = \mathcal{L}(G')$. □

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, добавяме нови променливи $A_1, \ldots, A_{k-2}$, и правила

Новата граматика ще има дължина $\mathcal{O}(|G|)$.

$$A \to X_1 A_1,\ A_1 \to X_2 A_2, \ldots,\ A_{k-2} \to X_{k-1} X_k.$$

4.8 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 4.3. 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 $A \to u_1u_2$, където $u_1, u_2 \in V \cup \Sigma$, заменяме всяка буква x с новата променлива U_x и добавяме правилото $U_x \to x$. Например, правилото $A \to aB$ се заменя с правилото $A \to U_aB$ и добавяме правилото $U_a \to a$, където U_a 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 $S \to \alpha$.

□

Забележка. Редът на стъпките е важен. Трябва да премахнем дългите правила преди ε-правилата. Вижте [9, стр. 296].

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 $\mathcal{L}(G') = \mathcal{L}(G) \cup \{\varepsilon\}$.

Теорема 4.4. 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 4.5. 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 4.3* 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 $G' = \langle \boldsymbol{V}', \boldsymbol{\Sigma}, \boldsymbol{S}, \boldsymbol{R}' \rangle$ строим граф с възли променливите от $\boldsymbol{V}'$ като за $A, B \in \boldsymbol{V}'$ имаме ребро $A \to B$ в графа точно тогава, когато съществува $C \in \boldsymbol{V}'$, 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 $\lambda, \rho \in (\boldsymbol{V} \cup \boldsymbol{\Sigma})^\star$, за които $A_0 \overset{n+1}{\lhd} \lambda A_0 \rho$ и $|\lambda\rho| = n + 1$

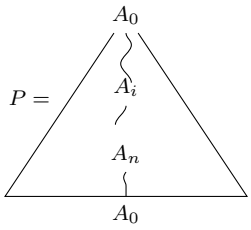
Понеже в G няма безполезни променливи, то $\lambda \lhd \alpha$, където $\alpha \in \boldsymbol{\Sigma}^\star$ и $|\lambda| \leq |\alpha|$ и $\rho \overset{\star}{\lhd} \gamma$, където $\gamma \in \boldsymbol{\Sigma}^\star$ и $|\rho| \leq |\gamma|$. Оттук заключаваме, че $A_0 \lhd \alpha A_0 \gamma$ и $|\alpha\gamma| \geq 1$. Понеже A_0 не е безполезна променлива, то $A_0 \lhd \beta$, за някоя дума $\beta \in \boldsymbol{\Sigma}^\star$. Сега комбинираме *Твърдение 4.3* и *Твърдение 4.4* за да получим следния извод:

$$\frac{A_0 \overset{\star}{\lhd} \alpha A_0 \gamma \qquad A_0 \overset{\star}{\lhd} \beta \qquad i \in \mathbb{N}}{A_0 \overset{\star}{\lhd} \alpha^i \beta \gamma^i}$$

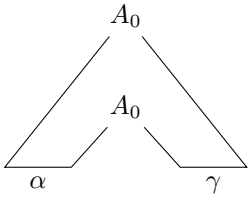
Понеже $|\alpha\beta| \geq 1$, заключаваме, че $\mathcal{L}(G)$ е безкраен език. Така доказахме, че ако в графът има цикли, то $\mathcal{L}(G)$ е безкраен език.

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 $k \leq |\boldsymbol{V}|$. Това означава, че ако $A \overset{\star}{\lhd} \alpha$, за някоя $\alpha \in \boldsymbol{\Sigma}^\star$, 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. □

(а) $A_0 \overset{n+1}{\lhd} \lambda A_0 \rho$, където $\lambda, \rho \in (\boldsymbol{V} \cup \boldsymbol{\Sigma})^\star$.



(б) $A_0 \overset{\geq n+1}{\lhd} \alpha A_0 \gamma$, където $\alpha, \gamma \in \boldsymbol{\Sigma}^\star$.



Да напомним, че всички от изброените тук проблеми са алгоритмично разрешими за регулярни езици. Вижда се, че за безконтекстни езици нещата не са толкова прости.

Проблемът за празнота: Да. *Следствие 4.6* ни казва, че съществува алгоритъм, който разрешава проблема за празнота за безконтекстни езици.

Проблемът за безкрайност: Да. От *Теорема 4.5* знаем, че този проблем е алгоритмично разрешим.

Проблемът за универсалност: Не. Чак в *Теорема 6.9* ще видим, че не съществува алгоритъм, който да разрешава този проблем за безконтекстни езици.

Проблемът за принадлежност: Да. Според *Теорема 4.6* проблемът за принадлежност е алгоритмично разрешим за безконтекстни езици.

Проблемът за включване: Не. Да фиксираме безконтекстна граматика G_0, за която $\mathcal{L}(G_0) = \boldsymbol{\Sigma}^\star$. Съобразете, че за произволна безконтекстна граматика G

имаме еквивалентността:

$$\mathcal{L}(G) = \mathbf{\Sigma}^\star \;\Leftrightarrow\; \mathcal{L}(G_0) \subseteq \mathcal{L}(G).$$

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, защото ако този проблем беше разрешим, то и проблемът за универсалност щеше да е разрешим.

4.9 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 4.6. Съществува *полиномиален* алгоритъм относно дължината на входната дума, който проверява дали дадена дума принадлежни на граматиката G.

Преди да достигнем до алгоритъм за проверка за принадлежност на дума към език на граматика, ще направим някои наблюдения.

Първо, понеже входната граматика е в нормална форма на Чомски, то имаме следното полезно свойство:

$A \lhd \alpha$ точно тогава, когато съществува правило $A \to BC$ и думата α може да се разбие като $\alpha = \beta\gamma$, където $B \lhd \beta$ и $C \lhd \gamma$.

Оттук веднага се вижда, че за да разберем дали една дума е в езика на граматиката, то трябва да знаем от кои променливи се извеждат различни по-къси части на тази дума.

За да бъдем по-конкретни, нека разгледаме входна дума $\alpha = a_0 a_1 \cdots a_{n-1}$ и да разгледаме множествата

$$\texttt{V[i][j]} = \{A \in \boldsymbol{V} \,:\, A \lhd_G a_i a_{i+1} \cdots a_j\},$$

където $0 \leq i \leq j < n$. Тогава $\alpha \in \mathcal{L}(G)$ точно тогава, когато $S \in \texttt{V[0][n-1]}$. Как можем да намерим тези множества?

Можем да пренапишем горното свойство по следния начин:

$A \in \texttt{V[i][j]}$ точно тогава, когато съществува правило $A \to BC$, съществува индекс k, $i \leq k < j$, за който $B \in \texttt{V[i][k]}$ и $C \in \texttt{V[k+1][j]}$.

Понеже по-дълги части от думата α се описват чрез по-къси части, то ще опишем как намираме множествата `V[i][j]` като постепенно увеличаваме разликата $j - i$.

- Нека $i = j$. Тогава за всяко $i < n$ е ясно, че:

$$\texttt{V[i][i]} = \{A \in V \,:\, A \to_G a_i\}.$$

- Нека $i < j$. Тогава за всяко k, където $i \leq k < j$, ако намерим правило $A \to BC$, за което $B \in$ `V[i][k]` и $C \in$ `V[k+1][j]`, то добавяме A към множеството `V[i][j]`.

  Множествата `V[i][k]` и `V[k+1][j]` са изчислени на предишни стъпки, защото $k - i < j - i$ и $j - (k+1) < j - i$.

Algorithm 5 Проблемът за принадлежност за безконтекстни езици

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], [21, стр. 141]. При вход дума α, алгоритъмът работи за време $\mathcal{O}(|\alpha|^3)$.

```
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 → 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 4.8. 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 5, е изпълнено, че за всяко $i < n - s$,

$$\texttt{V[i][i+s]} = \{A \in \boldsymbol{V} \,:\, A \lhd a_i a_{i+1} \cdots a_{i+s}\}.$$

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 $i < n$,

$$\texttt{V[i][i]} = \{A \in \boldsymbol{V} \,:\, A \to \alpha[i]\} = \{A \in \boldsymbol{V} \,:\, A \lhd a_i\}.$$

Сега ще докажем твърдението за $s > 0$, т.е. ще докажем, че за всяко $i < n - s$ е изпълнено, че:

$$\texttt{V[i][i+s]} = \{A \in \boldsymbol{V} \,:\, A \lhd a_i a_{i+1} \cdots a_{i+s}\}.$$

Да разгледаме двете посоки на това равенство.

($\subseteq$) Нека $A \in$ `V[i][i+s]`. Единствената стъпка на алгоритъма, при която може да сме добавили променливата A към множеството `V[i][i+s]` е ред (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 $i + s = (k + 1) + \underbrace{(i + s - k - 1)}_{s''}$;
- $A \to BC$ е правило в граматиката G.

Понеже $s' < s$ и $s'' < s$, от индукционното предположение имаме следното:

$$B \in \texttt{V[i][k]} \overset{\textbf{(И.П.)}}{\Longrightarrow} B \lhd a_i a_{i+1} \cdots a_k$$
$$C \in \texttt{V[k+1][i+2]} \overset{\textbf{(И.П.)}}{\Longrightarrow} C \lhd a_{k+1} \cdots a_{i+s}.$$

Заключаваме веднага, че $A \lhd a_i a_{i+1} \cdots a_{i+s}$.

($\supseteq$) Нека $A \lhd a_i a_{i+1} \cdots a_{i+s}$ и да разгледаме първото правило, което сме приложили в този извод. Понеже G е в НФЧ, правилото е от вида $A \to BC$ и тогава съществува k, за което $i \leq k < i + s$, и

$$B \lhd a_i \cdots a_k$$
$$C \lhd a_{k+1} \cdots a_{i+s}.$$

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]`.

□

Примери

Пример 4.7. Да разгледаме безконтекстната граматика

$$G = \begin{cases} S \to AB \mid aA \\ A \to a \mid aAa \\ B \to SB \mid BC \\ C \to \varepsilon \mid cC. \end{cases}$$

Първо да намерим променливите, от които се извеждат думи.

- $\texttt{Gen[0]} = \emptyset$;
- $\texttt{Gen[1]} = \{A, C\}$, защото $A \to a$ и $C \to \varepsilon$;
- $\texttt{Gen[2]} = \{A, C, S\}$, защото $S \to aA$;
- не можем да добавим B към `Gen[3]`, следователно $\texttt{Gen[3]} = \texttt{Gen[2]}$.

Получаваме граматиката

$$G' = \begin{cases} S \to aA \\ A \to a \mid aAa \\ C \to \varepsilon \mid cC. \end{cases}$$

Сега премахваме променливите и буквите, които не са достижими от началната промелива S.

- $\texttt{Reach[0]} = \{S\}$;
- $\texttt{Reach[1]} = \{S, A\}$, защото $S \to aAa$;
- $\texttt{Reach[2]} = \{S, A\}$.

Така получаваме граматиката

$$G'' = \begin{cases} S \to aA \\ A \to a \mid aAa. \end{cases}$$

Задача 4.16. Проверете дали $\mathcal{L}(G) = \emptyset$, където

$$G = \begin{cases} S \to AS \mid BC \\ A \to a \mid BA \mid SB \\ B \to b \mid BC \mid AB \\ C \to CB \mid SC \mid AS. \end{cases}$$

Упътване. Лесно се проверява, че:

$$\texttt{Gen[1]} = \{A, B\}$$
$$\texttt{Gen[2]} = \{A, B\}.$$

□

Пример 4.8. Нека е дадена граматиката

$$G = \begin{cases} S \to D \\ D \to AD \mid b \\ A \to AB \mid BC \mid a \\ B \to AA \mid UC \\ C \to \varepsilon \mid CA \mid a \\ U \to \varepsilon \mid aUb. \end{cases}$$

Тогава можем да намерим множеството от променливи E, от които се извежда ε в граматиката G като започнем да намираме неговите апроксимации:

- $\texttt{E[0]} \stackrel{\text{деф}}{=} \emptyset$;
- $\texttt{E[1]} = \{C, U\}$, защото $C \to \varepsilon$ и $U \to \varepsilon$;
- $\texttt{E[2]} = \{C, U, B\}$, защото $B \to UC$;
- $\texttt{E[3]} = \{C, U, B, A\}$, защото $A \to BC$;
- $\texttt{E[4]} = \texttt{E[3]}$.
- Оттук е ясно, че $(\forall n \geq 3)[\texttt{E[n]} = \texttt{E[3]}]$. Заключаваме, че
$$\texttt{E} = \{X \in V \mid X \lhd \varepsilon\} = \{C, U, B, A\}.$$

Това означава, че $\varepsilon \notin \mathcal{L}(G)$, защото $S \notin \texttt{E}$. Получаваме граматиката G' без ε-правила, за която $\mathcal{L}(G') = \mathcal{L}(G)$, където:

$$G' = \begin{cases} S \to D \\ D \to AD \mid D \mid b \\ A \to A \mid B \mid C \mid AB \mid BC \mid a \\ B \to A \mid C \mid AA \mid U \mid UC \\ C \to C \mid A \mid CA \mid a \\ U \to aUb \mid ab. \end{cases}$$

Пример 4.9. Нека е дадена граматиката

$$G = \begin{cases} S \to B \mid CC \mid b \\ A \to S \mid SB \\ B \to C \mid BC \\ C \to AB \mid a \mid b. \end{cases}$$

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 `Unit[3]` $=$ `Unit[4]`. Оттук можем да заключим следното:

$$A \lhd A,B,S,C$$
$$B \lhd B,C$$
$$S \lhd S,B,C$$
$$C \lhd C.$$

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 $A \lhd B,S,C$, то добавяме към R' правилата:
$$A \to BC \mid AB \mid a \mid b \mid CC.$$
- Понеже имаме,че $B \lhd_G C$, то добавяме към R' правилата:
$$B \to AB \mid a \mid b.$$
- Понеже имаме, че $S \lhd_G B,C$, то добавяме към R' правилата:
$$S \to BC \mid AB \mid a \mid b.$$

Накрая получаваме граматиката

$$G' = \begin{cases} S \to BC \mid AB \mid CC \mid a \mid b \\ A \to BS \mid BC \mid AB \mid a \mid b \mid CC \\ B \to AB \mid a \mid b \mid BC \\ C \to AB \mid a \mid b. \end{cases}$$

Задача 4.17. Премахнете преименуващите правила, като запазите езика, от граматиката

$$G = \begin{cases} S \to C \mid CC \mid b \\ A \to B \\ B \to S \mid C \mid BC \\ C \to a \mid AB; \end{cases}$$

Задача 4.18. Нека е дадена граматиката $G = \langle \{S, A, B, C\}, \{a, b\}, S, R \rangle$. Използвайте обща конструкция, за да премахнете „дългите" правила от G като при това получите безконтестна граматика G_1 с език $\mathcal{L}(G) = \mathcal{L}(G_1)$, където правилата на граматиката са:

$$S \to CC \mid b$$
$$A \to BSB \mid a$$
$$B \to ba \mid BC$$
$$C \to BaSA \mid a \mid b.$$

Пример 4.10. Нека е дадена безконтекстната граматика G в нормална форма на Чомски, където:

$$G = \begin{cases} S \to a \mid AB \mid AC \\ A \to a \\ B \to b \\ C \to SB \mid AS. \end{cases}$$

Ще приложим CYK алгоритъма за да проверим дали $aaabb \in \mathcal{L}(G)$.

- За $s = 0$ имаме, че:
  - $\texttt{V[0][0]} = \texttt{V[1][1]} = \texttt{V[2][2]} = \{S, A\}$;
  - $\texttt{V[3][3]} = \texttt{V[4][4]} = \{B\}$.
- За $s = 1$ имаме, че:
  - $\texttt{V[0][1]} = \texttt{V[1][2]} = \{C\}$;
  - $\texttt{V[2][3]} = \{S, C\}$;
  - $\texttt{V[3][4]} = \emptyset$.
- За $s = 2$ имаме, че:
  - $\texttt{V[0][2]} = \{S\} \cup \emptyset$;
  - $\texttt{V[1][3]} = \{S, C\} \cup \emptyset$;
  - $\texttt{V[2][4]} = \emptyset \cup \{C\}$.
- За $s = 3$ имаме, че:
  - $\texttt{V[0][3]} = \{S, C\} \cup \emptyset \cup \emptyset = \{S, C\}$;
  - $\texttt{V[1][4]} = \{S\} \cup \emptyset \cup \{C\} = \{S, C\}$.
- За $s = 4$ имаме, че:
  - $\texttt{V[0][4]} = \{S, C\} \cup \emptyset \cup \emptyset \cup \{C\} = \{S, C\}$.

Понеже $S \in \texttt{V[0][4]}$, то $aaabb \in \mathcal{L}(G)$.

Сега да видим защо $aba \notin \mathcal{L}(G)$.

- За $s = 0$ имаме, че:
  - $\texttt{V[0][0]} = \texttt{V[2][2]} = \{S, A\}$.
  - $\texttt{V[1][1]} = \{B\}$.
- За $s = 1$ имаме, че:
  - $\texttt{V[0][1]} = \{S, C\}$.
  - $\texttt{V[1][2]} = \emptyset$.
- За $s = 2$ имаме, че:
  - $\texttt{V[0][2]} = \{C\}$.

Понеже $S \notin \texttt{V[0][2]}$, то $aba \notin \mathcal{L}(G)$.

4.10 Операции върху езици

Задача 4.19. Докажете, че ако L е безконтекстен език, то $\texttt{Rev}(L)$ е безконтекстен.

Задача 4.20. Докажете, че ако L е безконтекстен език, то $\texttt{Pref}(L)$ е безконтекстен.

Задача 4.21. Докажете, че ако L е безконтекстен език, то $\omega^{-1}(L)$ е безконтекстен.

Знаем, че за всеки регулярен език L и произволен език M, $M^{-1}(L)$ е регулярен. Това е *Задача 2.18*, където е изложен алгебричен подход следвайки дефиницията на регулярните езици. Автоматният подход е *Задача 3.32*. [15, стр. 149]

Задача 4.22. Нека L е безконтекстен език, а R е регулярен език. Докажете, че езикът $R^{-1}(L)$ е безконтекстен.

Упътване. Най-лесно се вижда с декартова конструкция на стеков автомат за L и автомат за R. □

Триене на третина от думите на език

Да напомним следните операции върху езици:

$$\texttt{Triples}_2(L) \stackrel{\text{деф}}{=} \{\omega_2 \in \mathbf{\Sigma}^\star \,:\, (\exists\omega_1)(\exists\omega_3)[\,\omega_1\omega_2\omega_3 \in L \,\&\, |\omega_1| = |\omega_2| = |\omega_3|\,]\,\};$$

$$\texttt{Triples}_{1,3}(L) \stackrel{\text{деф}}{=} \{\omega_1\omega_3 \in \mathbf{\Sigma}^\star \,:\, (\exists\omega_2)[\omega_1\omega_2\omega_3 \in L \,\&\, |\omega_1| = |\omega_2| = |\omega_3|]\}.$$

Вече знаем, че $\texttt{Triples}_{1,3}(L)$ не винаги е регулярен при регулярен L. Възможно е за регулярен L да се конструира и стеков автомат за $\texttt{Triples}_{1,3}(L)$ [7, стр. 140].

Задача 4.23. Докажете, че ако L е регулярен език, 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:

- $S \to [(\boldsymbol{s}, q), (q, r), f]$ за произволно $f \in \boldsymbol{F}$.
- $[(p, q), (r, s), t] \to a[(p', q), (r', s), t']b$, където са изпълнени условията:
  - $\boldsymbol{\delta}(p, a) = p'$;
  - $\boldsymbol{\delta}(t', b) = t$, за някое t';
  - $\boldsymbol{\delta}(r, x) = r'$ за някое $x \in \Sigma$.
- $[(p, p), (q, q), q] \to \varepsilon$ за произволни $p, q \in \boldsymbol{Q}$.

□

Разполовяване на думи

Да напомним следната операция върху езици:

$$\texttt{Half}(L) \stackrel{\text{деф}}{=} \{\omega \in \mathbf{\Sigma}^\star \,:\, \omega \cdot \omega \in L\}.$$

Знаем, че ако един език L е регулярен, то $\texttt{Half}(L)$ също е регулярен. Интересно е да отбележим, че това свойство не се запазва за класа на безконтекстните езици.

Операцията $\texttt{Double}(L)$ не е толкова интересна, защото не е трудно да се намери безконтекстен език, за който $\texttt{Double}(L)$ не е безконтекстен.

Задача 4.24. Дайте пример за безконтекстен език L, за който $\texttt{Half}(L)$ *не* е безконтекстен.

Упътване. Разгледайте безконтекстния език

$$L = \{a^n b^n c^k a^k b^\ell c^\ell \,:\, n, k, \ell \geq 1\}.$$

Съобразете, че

$$\texttt{Half}(L) = \{a^n b^n c^n \ : \ n \geq 1\}.$$

□

Триене на половината дума

Да напомним сега още една операция върху езици:

$$\frac{1}{2}(L) \stackrel{\text{деф}}{=} \{\alpha \in \mathbf{\Sigma}^\star \ : \ (\exists \beta)[|\alpha| = |\beta| \ \& \ \alpha\beta \in L\}.$$

От *Задача 3.12* знаем, че ако един език L е регулярен, то $\frac{1}{2}(L)$ също е регулярен.

Задача 4.25. Дайте пример за безконтекстен език L, за който $\frac{1}{2}(L)$ *не* е безконтекстен.

Упътване. Например, разгледайте безконтекстния език

$$L = \{a^n b^n a^k b a^{3k+1} \ : \ n, k \in \mathbb{N}\}.$$

Тогава

$$\frac{1}{2}(L) \cap \mathcal{L}(\boldsymbol{a}^+\boldsymbol{b}^+\boldsymbol{a}^+\boldsymbol{b}) = \{a^n b^n a^n b \ : \ n \geq 1\},$$

който със сигурност не е безконтекстен. □

4.11 Задачи

Равен брой леви и десни скоби

Нека за по-голяма яснота да положим

За да се приближим малко до по-реален пример, можете да си мислите, че тук искаме да разпознаем думите с равен брой леви и десни скоби, като например интерпретираме a като символа { и b като символа }.

$$\texttt{left}(\alpha) \stackrel{\text{деф}}{=} |\alpha|_a \qquad \text{// брой срещания на } a \text{ в } \alpha$$
$$\texttt{right}(\alpha) \stackrel{\text{деф}}{=} |\alpha|_b \qquad \text{// брой срещания на } b \text{ в } \alpha$$

Задача 4.26. Нека ω е произволна дума над азбуката $\{a, b\}$. Тогава:

а) ако $\texttt{left}(\omega) = \texttt{right}(\omega)+1$, то съществуват думи ω_1, ω_2, за които е изпълнено:

- $\omega = \omega_1 a \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

б) ако $\texttt{right}(\omega) = \texttt{left}(\omega)+1$, то съществуват думи ω_1, ω_2, за които е изпълнено:

- $\omega = \omega_1 b \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

Упътване. Ще се съсредоточим върху случая, когато ω е дума, за която $\texttt{left}(\omega) = \texttt{right}(\omega) + 1$. Ще докажем а) с индукция по дължината на думата. Другият случай е аналогичен

- $|\omega| = 1$. Тогава $\omega_1 = \omega_2 = \varepsilon$ и $\omega = a$.
- Да приемем, че твърдението а) е изпълнено за всички думи ω с дължина $\leq n$.
- Нека сега $|\omega| = n+1$. 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 букви a и b, 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 4.27. 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 4.28. Докажете, че езикът

$$L = \{\, \omega \in \{a,b\}^\star \,:\, \texttt{left}(\omega) = \texttt{right}(\omega) \,\}$$

е безконтекстен.

Алтернативна граматика за езика L е

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

но уравнението за тази граматика има много решения.

Доказателство. Една възможна граматика G е следната:

$$S \to aSbS \mid bSaS \mid \varepsilon.$$

Като следствие от *Задача 4.26* може лесно да се изведе, че за думи ω, за които $\texttt{left}(\omega) = \texttt{right}(\omega)$, е изпълнено следното:

а) ако $\omega = a\omega'$, то са изпълнени свойствата:

- $\omega = a\omega_1 b \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

б) ако $\omega = b\omega'$, то са изпълнени свойствата:

- $\omega = b\omega_1 a \omega_2$;
- $\texttt{left}(\omega_1) = \texttt{right}(\omega_1)$;
- $\texttt{left}(\omega_2) = \texttt{right}(\omega_2)$.

Да напомним, че $\mathcal{L}^\ell_G(S) = \{\omega \in \Sigma^\star \,:\, S \overset{\leq \ell}{\lhd} \omega\}$ и според *Твърдение 4.5* имаме следната рекурсивна връзка:

$$\begin{aligned}
\mathcal{L}^0_G(S) =& \emptyset \\
\mathcal{L}^{\ell+1}_G(S) =& \{a\} \cdot \mathcal{L}^\ell_G(S) \cdot \{b\} \cdot \mathcal{L}^\ell_G(S) \cup \\
& \{b\} \cdot \mathcal{L}^\ell_G(S) \cdot \{a\} \cdot \mathcal{L}^\ell_G(S) \cup \\
& \{\varepsilon\}.
\end{aligned}$$

За коректност на граматиката трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}^\ell_G(S) \subseteq \{\omega \in \{a,b\}^\star \,:\, \texttt{left}(\omega) = \texttt{right}(\omega)\}. \tag{4.13}$$

Очевидно е, че *Свойство 4.13* е изпълнено за $\ell = 0$. Да приемем, че *Свойство 4.13* е изпълнено за някое ℓ. Ще докажем *Свойство 4.13* 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:

$$\mathcal{L}_G(S) = \bigcup_{\ell} \mathcal{L}_G^{\ell}(S) \subseteq \{\omega \in \{a,b\}^\star \,:\, \texttt{left}(\omega) = \texttt{right}(\omega)\}.$$

Сега за пълнота на граматиката трябва да докажем, че

$$\{\omega \in \{a,b\}^\star \,:\, \texttt{left}(\omega) = \texttt{right}(\omega)\} \subseteq \mathcal{L}_G(S). \tag{4.14}$$

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 4.26* 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:

$$\{\omega \in \{a,b\}^\star \,:\, \texttt{left}(\omega) = \texttt{right}(\omega)\} \subseteq \mathcal{L}_G(S).$$

□

Балансирани скоби

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 }.

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, $\texttt{left}(\gamma) \geq \texttt{right}(\gamma)$.

Задача 4.29. Докажете, че

$$L = \{\ \alpha \in \{a,b\}^\star\ :\ \alpha \text{ е балансирана дума }\}$$

е безконтекстен език.

[13, стр. 135]
✍ Докажете, че езикът L не е регулярен!
Алтернативна граматика е

$$S \to aSbS \mid \varepsilon.$$

Тя е по-добра в смисъл, че системата за тази граматика има единствено решение.

Доказателство. Да разгледаме граматиката G с правила

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

Ще докажем, че $L = \mathcal{L}(G)$. Имаме, че

$$\begin{aligned}\mathcal{L}_G^0(S) =& \emptyset \\ \mathcal{L}_G^{\ell+1}(S) =& \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{b\} \cup \\ & \mathcal{L}_G^\ell(S) \cdot \mathcal{L}_G^\ell(S)\ \cup \{\varepsilon\}.\end{aligned}$$

Първо да разгледаме коректност на граматиката, т.е. трябва да докажем, че за всяко ℓ е изпълнено следното:

$$\mathcal{L}_G^\ell(S) \subseteq \{\ \alpha \in \{a,b\}^\star\ :\ \alpha \text{ е балансирана дума }\}. \tag{4.15}$$

Твърдението е очевидно за $\ell = 0$. Да приемем, че *Свойство 4.15* е изпълнено за някое ℓ. Ще докажем *Свойство 4.15* за $\ell + 1$. Да разгледаме произволна дума $\omega \in \mathcal{L}_G^{\ell+1}(S)$. Имаме три случая.

- Нека $\omega = \varepsilon$. Тогава е ясно, че ω е балансирана дума.
- Нека $\omega \in \{a\} \cdot \mathcal{L}_G^\ell(S) \cdot \{b\}$. Тогава $\omega = a\omega_1 b$ и $\omega_1 \in \mathcal{L}_G^\ell(S)$. От **(И.П.)** имаме, че ω_1 е балансирана. Лесно се съобразява, че ω също е балансирана.
- Нека $\omega \in \mathcal{L}_G^\ell(S) \cdot \mathcal{L}_G^\ell(S)$. Тогава $\omega = \omega_1\omega_2$, такива че $\omega_1, \omega_2 \in \mathcal{L}_G^\ell(S)$. От **(И.П.)** имаме, че ω_1 и ω_2 са балансирани. Лесно се съобразява, че ω също е балансирана.

Така доказахме, че

$$\mathcal{L}_G(S) = \bigcup_\ell \mathcal{L}_G^\ell(S) \subseteq \{\ \alpha \in \{a,b\}^\star\ :\ \alpha \text{ е балансирана дума }\}.$$

Сега ще докажем пълнота на граматиката, т.е. следното свойство:

$$\{\, \alpha \in \{a,b\}^\star \,:\, \alpha \text{ е балансирана дума } \} \subseteq \mathcal{L}_G(S). \tag{4.16}$$

Това ще направим с *пълна* индукция по дължината на думите. Да разгледаме произволна балансирана дума ω. Имаме няколко случая, които трябва да разгледаме.

- Ако $\omega = \varepsilon$, то е ясно, че $\omega \in \mathcal{L}_G(S)$.
- Нека сега $\omega \neq \varepsilon$. Ясно е, че със сигурност $\omega = a\omega_1 b$. Проблемът е, че в общия случай не е ясно дали можем да приложим **(И.П.)** за ω_1, защото е възможно ω_1 да не е балансирана. Например, ако $\omega = abab$, то $\omega_1 = ba$ не е балансирана. Поради тази причина, трябва да сме по-внимателни и да разгледаме два допълнителни случая.
  - Нека ω има *същински* префикс ω_1, който да е балансирана дума. Понеже ω е балансирана дума, лесно се съобразява, че $\omega = \omega_1\omega_2$, ω_2 също е балансирана дума. Сега можем да приложим **(И.П.)** за ω_1 и ω_2 и да получим, че $\omega_1 \in \mathcal{L}_G(S)$ и $\omega_2 \in \mathcal{L}_G(S)$. Ясно е, че $\omega \in \mathcal{L}_G(S)$.

    Тук $\omega_1 \neq \varepsilon$ и $\omega_1 \neq \alpha$. Например, ab е същински префикс на $abab$, който е балансирана дума.
  - Нека ω да няма същински префикс ω_1, който е балансирана дума. Ясно е, че тогава $\omega = a\beta b$, за някое β. Да видим защо β е балансирана дума. Ако β е балансирана, то ще можем да приложим **(И.П.)** за β и ще сме готови.

    Например, $aabb$ няма същински префикс, който да е балансирана дума.

    За всеки префикс γ на β имаме, че $a\gamma$ е префикс на α, и понеже α е балансирана, то $\texttt{left}(a\gamma) \geq \texttt{right}(a\gamma)$. Възможно ли е $\texttt{left}(\gamma) < \texttt{right}(\gamma)$? Това може да се случи единствено ако $\texttt{left}(a\gamma) = \texttt{right}(a\gamma)$. Но тогава $a\gamma$ е същински префикс на α, за който $a\gamma$ е балансирана дума, което противоречи на случая, който разглеждаме. Това означава, че за произволен префикс γ на β, $\texttt{left}(\gamma) \geq \texttt{right}(\gamma)$ и оттук β е балансирана дума и можем да приложим **(И.П.)** . Тогава $\beta \in \mathcal{L}_G(S)$ и следователно $\alpha \in \{a\} \cdot \mathcal{L}_G(S) \cdot \{b\} \subseteq \mathcal{L}_G(S)$.

Така доказахме *Свойство 4.16*, което ни дава пълнота на граматиката спрямо езика. □

Лесни задачи

Задача 4.30. Постройте регулярен израз за езика на следната граматика:

$$S \to S + S \mid S * S \mid A$$
$$A \to KL \mid LK$$
$$K \to 0K \mid \varepsilon$$
$$L \to 1K \mid \varepsilon.$$

Задача 4.31. Докажете, че следните езици са безконтекстни.

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) $S \to EaE$, $E \to aEbE \mid bEaE \mid \varepsilon$;

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 4.32. Проверете дали следните езици са безконтекстни:

а) $\{a^n b^{2n} c^{3n} \mid n \in \mathbb{N}\}$;

б) $\{a^n b^k c^k a^n \mid k \leq n\}$;

в) $\{a^n b^m c^k \mid n < m < k\}$;

г) $\{a^n b^n c^k \mid n \leq k \leq 2n\}$;

д) $\{a^n b^m c^k \mid k = \min\{n, m\}\}$;

е) $\{a^n b^n c^m \mid m \leq n\}$;

ж) $\{a^n b^m c^k \mid k = n \cdot m\}$;

з) $L^\star$, където $L = \{\alpha\alpha^{\mathtt{rev}} \mid \alpha \in \{a,b\}^\star\}$;

и) $\{\omega\omega\omega \mid \omega \in \{a,b\}^\star\}$;

к) $\{a^{n^2} b^n \mid n \in \mathbb{N}\}$;

л) $\{a^p \mid p$ е просто $\}$;

м) $\{\omega \in \{a,b\}^\star \mid \omega = \omega^{\mathtt{rev}}\}$;

н) $\{\omega^n \mid \omega \in \{a,b\}^\star \,\&\, |\omega|_b = 2 \,\&\, n \in \mathbb{N}\}$;

о) $\{\omega c^n \omega^{\mathtt{rev}} \mid \omega \in \{a,b\}^\star \,\&\, n = |\omega|\}$;

п) $\{\alpha c \beta \mid \alpha, \beta \in \{a,b\}^\star \,\&\, \alpha$ е подниз на $\beta\}$;

р) $\{\omega_1 \sharp \omega_2 \sharp \ldots \sharp \omega_k \mid k \geq 2 \,\&\, \omega_i \in \{a\}^\star \,\&\, (\exists i,j)[i \neq j \,\&\, \omega_i = \omega_j]\}$;

с) $\{\omega_1 \sharp \omega_2 \sharp \ldots \sharp \omega_k \mid k \geq 2 \,\&\, \omega_i \in \{a\}^\star \,\&\, (\forall i, j \leq k)[i \neq j \;\Leftrightarrow\; \omega_i \neq \omega_j]\}$;

т) $\{a^i b^j c^k \mid i,j,k \geq 0 \,\&\, (i = j \vee j = k)\}$;

у) $\{\omega \in \{a,b,c\}^\star \mid |\omega|_a > |\omega|_b > |\omega|_c\}$;

ф) $\{a,b\}^\star \setminus \{a^n b^n \mid n \in \mathbb{N}\}$;

х) $\{a^n b^m c^k \mid m^2 = 2nk\}$;

ц) $L = \{a^n b^m c^m a^n \mid m,n \in \mathbb{N} \,\&\, n = m + 42\}$;

ч) $L = \{\sharp a \sharp aa \sharp aaa \sharp \cdots \sharp a^{n-1} \sharp a^n \sharp \mid n \geq 1\}$;

ш) $\{a^m b^n c^k \mid m = n \vee n = k \vee m = k\}$;

щ) $\{a^m b^n c^k \mid m \neq n \vee n \neq k \vee m \neq k\}$;

ю) $\{\omega \in \{a,b,c\}^\star \mid |\omega|_a \neq |\omega|_b \vee |\omega|_a \neq |\omega|_c \vee |\omega|_b \neq |\omega|_c\}$.

Не толкова лесни задачи

Задача 4.33. Докажете, че езикът $L = \{a^n b^{kn} \ : \ k, n > 0\}$ 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 4.34. За всеки от следните езици, отговорете дали са безконтекстни, като се обосновете:

а) $\{\alpha\sharp\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta^{\texttt{rev}}) = 1\}$; Да

б) $\{\alpha\sharp\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta^{\texttt{rev}}) \geq 1\}$; Да

в) $\{\alpha\sharp\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) \geq 1\}$; Не

г) $\{\alpha\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) \geq 1\}$; Да

д) $\{\alpha\beta \,:\, \alpha, \beta \in \{a,b\}^\star \,\&\, |\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) = 1\}$; Не

Задача 4.35. Да разгледаме езика

$$D_1 \stackrel{\text{деф}}{=} \{\alpha_1\sharp\cdots\sharp\alpha_n \,:\, n \geq 2 \,\&\, |\alpha_i| = |\alpha_{i+1}| \,\&\, \texttt{diff}(\alpha_i, \alpha_{i+1}^{\texttt{rev}}) = 1\}.$$

- 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

$$L_1 = \{\alpha_1\sharp\alpha_2 \,:\, |\alpha_1| = |\alpha_2| \,\&\, \texttt{diff}(\alpha_1, \alpha_2^{\texttt{rev}}) = 1\}.$$

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 4.36. За произволен език L, дефинираме езика

$$\texttt{Diff}_n(L) = \{\alpha \in L \,:\, (\exists \beta \in L)[|\alpha| = |\beta| \,\&\, \texttt{diff}(\alpha, \beta) = n]\}.$$

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?

[22, стр. 158]

Задача 4.37. Нека L_1 и L_2 са езици. Дефинираме

$$L_1 \triangle L_2 \stackrel{\text{деф}}{=} \{\alpha\beta \,:\, \alpha \in L_1 \,\&\, \beta \in L_2 \,\&\, |\alpha| = |\beta|\}.$$

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 4.38. 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 4.39. Да разгледаме езиците:

$$P = \{\alpha \in \{a, b, c\}^* \,:\, \alpha \text{ е палиндром с четна дължина}\}$$
$$L = \{\beta b^n \,:\, n \in \mathbb{N}, \beta \in P^n\}.$$

Да се докаже, че:

а) L не е регулярен;

б) L е безконтекстен.

Задача 4.40. 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:

$$L = \{\alpha_1 \cdots \alpha_{3n}\beta_1 \cdots \beta_m\gamma_1 \cdots \gamma_n \,:\, \alpha_i, \gamma_j \in L_1, \beta_k \in L_2, m, n \in \mathbb{N}\}.$$

Задача 4.41. Нека $L = \{\omega \in \{a,b\}^\star \,:\, |\omega|_a = 2\}$. Да се докаже, че езикът $L' = \{\alpha^n \,:\, \alpha \in L, n \geq 0\}$ не е безконтекстен.

Задача 4.42. Нека $\Sigma = \{a,b,c\}$ и $L \subseteq \Sigma^\star$ е безконтекстен 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 4.43. Докажете, че езикът

$$L = \{\, \omega_1\sharp\omega_2\sharp\cdots\sharp\omega_{2n} \,:\, n \geq 1 \;\&\; \sum_{i=1}^{n} |\omega_{2i-1}| = \sum_{i=1}^{n} |\omega_{2i}| \,\}$$

е безконтекстен.

Упътване. Най-лесно става със стеков автомат, като са необходими две допълнителни букви за азбуката на стека.

Една възможна безконтекстна граматика е следната:

$$\begin{aligned} &E \to XEX \mid \sharp O \mid O\sharp \mid \sharp E\sharp \\ &O \to OO \mid EE \mid \varepsilon \\ &X \to a \mid b, \end{aligned}$$

където началната променлива е E. □

Глава 5

Стекови автомати

5.1 Недетерминирани стекови автомати

Недетерминиран стеков автомат е седморка от вида

$$P = \langle Q, \boldsymbol{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle,$$

където:

- $\boldsymbol{Q}$ е крайно множество от състояния;
- $\boldsymbol{\Sigma}$ е крайна входна азбука;
- Γ е крайна стекова азбука;
- $\sharp \in \Gamma$ е символ за дъно на стека;
- $\boldsymbol{\Delta} : \boldsymbol{Q} \times \boldsymbol{\Sigma}_\varepsilon \times \Gamma \to \mathscr{P}(\boldsymbol{Q} \times \Gamma^{\leq 2})$ е функция на преходите;
- $\boldsymbol{s} \in \boldsymbol{Q}$ е начално състояние;
- $\boldsymbol{f} \in \boldsymbol{Q}$ е заключителното състояние.

На англ. *Push-down automaton*. В този курс няма да разглеждаме детерминирани стекови автомати. Когато кажем стеков автомат, ще имаме предвид недетерминиран стеков автомат. Означаваме

$$\Sigma_\varepsilon \stackrel{\text{деф}}{=} \Sigma \cup \{\varepsilon\}$$
$$\Gamma^{\leq 2} \stackrel{\text{деф}}{=} \{\varepsilon\} \cup \Gamma \cup \Gamma^2.$$

Обикновено се взима Δ функцията да има сигнатура $\boldsymbol{\Delta} : \boldsymbol{Q} \times \boldsymbol{\Sigma}_\varepsilon \times \Gamma \to \mathscr{P}(\boldsymbol{Q} \times \Gamma^\star)$. 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 $(q, \alpha, \gamma) \in \boldsymbol{Q} \times \boldsymbol{\Sigma}^\star \times \Gamma^\star$, 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 $\boldsymbol{Q} \times \boldsymbol{\Sigma}^\star \times \Gamma^\star$, която ще ни казва как моментното описание на автомата P се променя след изпълнение на една стъпка.

$$\frac{\Delta(q, x, A) \ni (p, \beta) \qquad \alpha \in \mathbf{\Sigma}^\star \qquad \rho \in \Gamma^\star}{(q, x\alpha, A\rho) \vdash_P (p, \alpha, \beta\rho)}$$

Фигура 5.1: Правило за извършване на една стъпка в изчисление на стеков автомат

От дефиницията на релацията $\vdash_P$ веднага получаваме свойството, че можем да добавяме произволни остатъци към входната дума и стека.

Твърдение 5.1. $$\frac{(q, \alpha, \gamma) \vdash_P (q', \alpha', \gamma') \qquad \beta \in \Sigma^\star \qquad \delta \in \Gamma^\star}{(q, \alpha\beta, \gamma\delta) \vdash_P (q', \alpha'\beta, \gamma'\delta)}$$

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 5.2: Дефиниция на релацията $\vdash_P^\ell$

Задача 5.1. Докажете, че за произволни $\ell_1 \geq 0$ и $\ell_2 \geq 0$,

$$\frac{\kappa_0 \vdash_P^{\ell_1} \kappa_1 \qquad \kappa_1 \vdash_P^{\ell_2} \kappa_2}{\kappa_0 \vdash_P^{\ell_1+\ell_2} \kappa_2}$$

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 5.1. Езикът $\mathcal{L}(P)$, който се разпознава от стековия автомат P дефинираме по следния начин:

$$\mathcal{L}(P) \stackrel{\text{деф}}{=} \{\, \omega \in \mathbf{\Sigma}^\star \,:\, (\boldsymbol{s}, \omega, \sharp) \vdash_P^\star (\boldsymbol{f}, \varepsilon, \varepsilon) \,\}.$$

Възможно е да се дадат и други еквивалентни дефиниции на $\mathcal{L}(P)$. В повечето учебници се прави разлика между езици, чиито думи се разпознават с финално състояние и такива с празен стек. След това се доказва, че двата вида езици съвпадат. За нашите цели това е излишно.

Твърдение 5.2. За всяко $\ell \geq 0$ е изпълнен извода:

$$\frac{(q, \alpha, \gamma) \vdash^\ell (p, \varepsilon, \varepsilon) \qquad \beta \in \mathbf{\Sigma}^\star \qquad \rho \in \Gamma^\star}{(q, \alpha\beta, \gamma\rho) \vdash^\ell (p, \beta, \rho)}$$

Доказателство.

- Случаят, когато $\ell = 0$ е ясен, защото тогава $q = p$, $\alpha = \varepsilon$ и $\gamma = \varepsilon$.
- Нека $\ell > 0$. Тогава имаме следния извод:

$$\frac{(q, \alpha, \gamma) \vdash (q', \alpha', \gamma') \qquad (q', \alpha', \gamma') \vdash^{\ell-1} (p, \varepsilon, \varepsilon)}{(q, \alpha, \gamma) \vdash^\ell (p, \varepsilon, \varepsilon)}$$

Тогава за произволни $\beta \in \Sigma^\star$ и $\rho \in \Gamma^\star$ получаваме следното:

$$\text{(Тв. 5.1)}\ \frac{\dfrac{(q, \alpha, \gamma) \vdash (q', \alpha', \gamma')}{(q, \alpha\beta, \gamma\rho) \vdash (q', \alpha'\beta, \gamma'\rho)} \qquad \dfrac{(q', \alpha', \gamma') \vdash^{\ell-1} (p, \varepsilon, \varepsilon)}{(q', \alpha'\beta, \gamma'\rho) \vdash^{\ell-1} (p, \beta, \rho)}}{(q, \alpha\beta, \gamma\rho) \vdash^\ell (p, \beta, \rho)}\ \textbf{(И.П.)}$$

□

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.

Твърдение 5.3. За произволни $\ell_1 \geq 0$ и $\ell_2 \geq 0$ е изпълнено:

$$\frac{(p, \alpha, A) \vdash_P^{\ell_1} (r, \varepsilon, \varepsilon) \qquad (r, \beta, B) \vdash_P^{\ell_2} (q, \varepsilon, \varepsilon)}{(p, \alpha\beta, AB) \vdash_P^{\ell_1+\ell_2} (q, \varepsilon, \varepsilon)}$$

Упътване. Достатъчно е да проследим извода:

$$\text{(Тв. 5.2)}\ \frac{\dfrac{(p,\alpha,A)\vdash_P^{\ell_1}(r,\varepsilon,\varepsilon)}{(p,\alpha\beta,AB)\vdash_P^{\ell_1}(r,\beta,B)} \qquad (r,\beta,B)\vdash_P^{\ell_2}(q,\varepsilon,\varepsilon)}{(p,\alpha\beta,AB)\vdash_P^{\ell_1+\ell_2}(q,\varepsilon,\varepsilon)}$$

□

Второто свойство ни казва кога можем да разбием едно изчисление на по-малки части. Това свойство ще бъде важно по-нататък.

Твърдение 5.4. 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. Понеже в стека имаме поне един символ, 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 $\alpha=x$ и $r=p$. Получаваме, че в този случай изчислението се разбива на 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 имаме изчислението $(q',\alpha',BC\rho)\vdash_P^{\ell-1}(p,\varepsilon,\varepsilon)$, 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 $\beta_1\beta_2=\alpha'$. Щом $\ell_2'<\ell$, пак 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:

- $(q, \underbrace{x\beta_1\gamma_1}_{\alpha_1}, A) \vdash_P^{1+\ell'_1+k_1} (r'', \varepsilon, \varepsilon)$;
- $(r'', \underbrace{\gamma_2}_{\alpha_2}, \rho) \vdash_P^{k_2} (p, \varepsilon, \varepsilon)$.

- Вече свършихме тежката работа. Нека 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:
  - $(q, \underbrace{x\alpha'_1}_{\alpha_1}, A) \vdash_P^{1+\ell'_1} (r', \varepsilon, \varepsilon)$;
  - $(r', \underbrace{\alpha'_2}_{\alpha_2}, \rho) \vdash_P^{\ell'_2} (p, \varepsilon, \varepsilon)$.
- Ако първата стъпка в изчислението е направена заради $\Delta(q, x, A) \ni (q', \varepsilon)$, то нека $\alpha = x\alpha'$. Тогава 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 5.1. От *Пример 4.5* знаем, че езикът

$$L = \{a^n b^n \ : \ n \in \mathbb{N}\}$$

е безконтекстен. Този език се разпознава и от стековия автомат

$$P = \langle Q, \mathbf{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle,$$

където:

- $Q \stackrel{\text{деф}}{=} \{q, p, f\}$;
- $\boldsymbol{s} \stackrel{\text{деф}}{=} q$ и $\boldsymbol{f} \stackrel{\text{деф}}{=} f$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$ 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 ε
  (4) $\Delta(q, b, a) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}$; // Започваме да четем b
  (5) $\Delta(p, b, a) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}$; // Чистим a 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 a^2b^2 се разпознава от 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 $(\boldsymbol{s}, a^2b^2, \sharp) \vdash_P^\star (\boldsymbol{f}, \varepsilon, \varepsilon)$, 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:

$$\begin{aligned}
(q, a^n\beta, \sharp) &\vdash_P^n (q, \beta, a^n\sharp)\\
(p, b^n, a^n\sharp) &\vdash_P^n (p, \varepsilon, \sharp).
\end{aligned}$$

Заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете, че с индукция по n, че за всяко естествено число n са изпълнени свойствата:

$$\begin{aligned}
(q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \gamma\sharp) &\implies \alpha = \gamma = a^n\\
(p, \beta, \gamma\sharp) \vdash_P^n (p, \varepsilon, \sharp) &\implies \beta = b^n \,\&\, \gamma = a^n.
\end{aligned}$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

Пример 5.2. От *Задача 4.2* знаем, че езикът

$$L = \{\, \omega\omega^{\mathtt{rev}} \ : \ \omega \in \{a, b\}^\star \,\}$$

е безконтекстен. Този език се разпознава и от стеков автомат

$$P = \langle Q, \mathbf{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle,$$

където:

- $Q \stackrel{\text{деф}}{=} \{q, p, f\}$ и $\boldsymbol{s} \stackrel{\text{деф}}{=} q$, $\boldsymbol{f} \stackrel{\text{деф}}{=} f$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b\}$, $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$;
- Функцията на преходите Δ има следната дефиниция:
  (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) $\Delta(p, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(f, \varepsilon)\}$;

  За всички липсващи твойки в дефиницията на Δ приемаме, че Δ връща $\emptyset$.

Основното наблюдение, което трябва да направим за да разберем конструкцията на автомата е, че всяка дума от вида $\omega\omega^{\mathtt{rev}}$ може да се запише като $\omega_1 aa\omega_1^{\mathtt{rev}}$ или $\omega_1 bb\omega_1^{\mathtt{rev}}$. Да видим защо P разпознава думата $abaaba$ с празен стек. Започваме по следния начин:

$$\begin{aligned}
(q, abaaba, \sharp) &\vdash_P (q, baaba, a\sharp) && \text{// правило (1)}\\
&\vdash_P (q, aaba, ba\sharp) && \text{// правило (5)}\\
&\vdash_P (q, aba, aba\sharp). && \text{// правило (4)}
\end{aligned}$$

Сега можем да направим два избора как да продължим. Състоянието p служи за маркер, което ни казва, че вече сме започнали да четем $\omega^{\mathtt{rev}}$. Поради тази причина, продължаваме така:

$$\begin{aligned}(q, aba, aba\sharp) &\vdash_P (p, ba, ba\sharp) && \text{// правило (3)}\\ &\vdash_P (p, a, a\sharp) && \text{// правило (6)}\\ &\vdash_P (p, \varepsilon, \sharp) && \text{// правило (6)}\\ &\vdash_P (f, \varepsilon, \varepsilon). && \text{// правило (7)}\end{aligned}$$

Да проиграем още един пример. Да видим защо думата aba не се извежда от автомата.

$$\begin{aligned}(q, aba, \sharp) &\vdash_P (q, ba, a\sharp) && \text{// правило (1)}\\ &\vdash_P (q, a, ba\sharp) && \text{// правило (5)}\\ &\vdash_P (q, \varepsilon, aba\sharp). && \text{// правило (4)}\end{aligned}$$

От последното моментно описание на автомата нямаме нито един преход, следователно думата aba не се разпознава от P.

а) Докажете с индукция по n, за всяко естествено число n са изпълнени свойствата:

$$|\alpha| = n \implies (q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \alpha^{\mathtt{rev}}\sharp) \tag{5.1}$$
$$|\beta| = n \implies (p, \beta, \beta\sharp) \vdash_P^n (p, \varepsilon, \sharp). \tag{5.2}$$

Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете с индукция по n, че за всяко естествено число n са изпълнени свойствата:

$$(q, \alpha\beta, \sharp) \vdash_P^n (q, \beta, \gamma\sharp) \implies \gamma = \alpha^{\mathtt{rev}} \,\&\, |\alpha| = n \tag{5.3}$$
$$(p, \beta, \gamma\sharp) \vdash_P^n (p, \varepsilon, \sharp) \implies \gamma = \beta \,\&\, |\beta| = n. \tag{5.4}$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

Пример 5.3. Знаем от *Задача 4.28*, че езикът

$$L = \{\, \omega \in \{a, b\}^\star \,:\, |\omega|_a = |\omega|_b \}$$

е безконтекстен. Този език се разпознава и от стековия автомат

$$P = \langle Q, \boldsymbol{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle,$$

където:

- $Q = \{q, f\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, b, \sharp\}$;
- $\boldsymbol{s} = q$ и $\boldsymbol{f} = f$;
- Да напомним, че от *Задача 4.28* знаем, че този език е безконтекстен.
- 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:

$$(\forall n)[a^n\gamma \in L \implies (q, \gamma, a^n\sharp) \vdash_P^\star (q, \varepsilon, \sharp)] \tag{5.5}$$
$$(\forall n)[b^n\gamma \in L \implies (q, \gamma, b^n\sharp) \vdash_P^\star (q, \varepsilon, \sharp)]. \tag{5.6}$$

Ще докажем едновременно *Свойство 5.5* и *Свойство 5.6* 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 5.6* 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 5.5* с думата γ' и получаваме, че:

$$\begin{aligned}(q, \overbrace{a^k b\gamma'}^{\gamma}, a^n\sharp) &\vdash^\star_P (q, b\gamma', a^{n+k}\sharp) && \text{// правило (3)}\\ &\vdash^\star_P (q, \gamma', a^{n+k-1}\sharp) && \text{// правило (5)}\\ &\vdash^\star_P (q, \varepsilon, \sharp). && \text{// \textbf{(И.П.)}}\end{aligned}$$

Аналогично се доказва и *Свойство 5.6*. Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

б) Докажете с пълна индукция по дължината на думата γ, че за произволна дума $\gamma \in \{a,b\}^\star$ е изпълнено, че:

$$(\forall n)[\,(q,\gamma,a^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp) \implies a^n\gamma \in L\,] \qquad (5.7)$$

$$(\forall n)[\,(q,\gamma,b^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp) \implies b^n\gamma \in L\,]. \qquad (5.8)$$

Нека имаме изчислението $(q,\gamma,a^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp)$. Да представим думата γ като $\gamma = a^k b\gamma'$.

- Ако $n+k=0$, то можем да разбием изчислението по следния начин:

$$\begin{aligned}(q, b\gamma', \sharp) &\vdash_P (q, \gamma', b\sharp)\\ &\vdash^\star (q, \varepsilon, \sharp).\end{aligned}$$

Тогава от **(И.П.)** за *Свойство 5.8* следва, че $a^n\gamma = b\gamma' \in L$.

- Ако $n+k>0$, то можем да разбием изчислението по следния начин:

$$\begin{aligned}(q, a^k b\gamma', a^n\sharp) &\vdash^\star_P (q, b\gamma', a^{n+k}\sharp)\\ &\vdash_P (q, \gamma', a^{n+k-1}\sharp)\\ &\vdash^\star (q, \varepsilon, \sharp).\end{aligned}$$

Тогава от **(И.П.)** за *Свойство 5.7* 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 5.8*. Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

Пример 5.4. От *Задача 4.29* знаем, че езикът

$$L = \{\, \omega \in \{a,b\}^\star \,:\, \omega \text{ е балансирана дума } \}$$

е безконтекстен. Този език се разпознава и от стековия автомат

$$P = \langle Q, \mathbf{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f}\rangle,$$

където:

- $Q = \{q, f\}$;
- $\boldsymbol{s} = q$ и $\boldsymbol{f} = f$;
- $\Sigma = \{a,b\}$ и $\Gamma = \{a, \sharp\}$;
- 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) $\Delta(q,b,a) = \{(q,\varepsilon)\}$;

(а) Докажете, че за произволно естествено число n и произволна дума $\beta \in \{a,b\}^\star$, е изпълнено, че:

$$a^n\beta \in L \implies (q,\beta,a^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp).$$

Оттук заключете, че $L \subseteq \mathcal{L}(P)$.

(б) Докажете, че за произволно естествено число n и произволна дума $\beta \in \{a,b\}^\star$, е изпълнено, че:

$$(q,\beta,a^n\sharp) \vdash^\star_P (q,\varepsilon,\sharp) \implies a^n\beta \in L.$$

Оттук заключете, че $\mathcal{L}(P) \subseteq L$.

5.2 Теорема за еквивалентност

Лема 5.1. За всяка безконтекстна граматика G, съществува стеков автомат P, такъв че $\mathcal{L}(G) = \mathcal{L}(P)$.

Доказателство. Нека е дадена произволна безконтекстна граматика $G = \langle \boldsymbol{V}, \boldsymbol{\Sigma}, \boldsymbol{R}, \boldsymbol{S} \rangle$ в нормална форма на Чомски. Нашата цел е да построим стеков автомат

$$P = \langle Q, \boldsymbol{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle,$$

който разпознава езика $\mathcal{L}(G)$.

- $\boldsymbol{Q} \stackrel{\text{деф}}{=} \{\boldsymbol{s}, p, \boldsymbol{f}\}$;
- $\Gamma \stackrel{\text{деф}}{=} \boldsymbol{\Sigma} \cup \boldsymbol{V} \cup \{\sharp\}$;
- Функцията на преходите $\boldsymbol{\Delta}$ за стековия автомат P дефинираме по следния начин:

  (1) $\boldsymbol{\Delta}(\boldsymbol{s}, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(p, S\sharp)\}$;

  (2) За всяка променлива $A \in \boldsymbol{V}$,

  $$\boldsymbol{\Delta}(p, \varepsilon, A) \stackrel{\text{деф}}{=} \{(p, \alpha) \ : \ A \to \alpha \text{ е правило в } G\}.$$

  (3) За всяка буква $a \in \boldsymbol{\Sigma}$,

  $$\boldsymbol{\Delta}(p, a, a) \stackrel{\text{деф}}{=} \{(p, \varepsilon)\}.$$

  (4) $\boldsymbol{\Delta}(p, \varepsilon, \sharp) \stackrel{\text{деф}}{=} \{(\boldsymbol{f}, \varepsilon)\}$.

Ще докажем, че за всяка променлива $A \in \boldsymbol{V}$ и всяка дума $\alpha \in \boldsymbol{\Sigma}^\star$, е изпълнено:

(а) ако $A \lhd \alpha$, то $(p, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon)$;

(б) ако $(p, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon)$, то $A \lhd \alpha$.

Ако приемем, че (а) и (б) са изпълнени, тогава, ако вземем $A = S$, то ще получим следната верига от еквивалентности:

$$\begin{aligned}
\alpha \in \mathcal{L}(G) &\Leftrightarrow S \lhd \alpha \\
&\Leftrightarrow (p, \alpha, S) \vdash_P^\star (p, \varepsilon, \varepsilon) && \text{// от (а) и (б)} \\
&\Leftrightarrow (\boldsymbol{s}, \alpha, \sharp) \vdash_P^\star (\boldsymbol{f}, \varepsilon, \varepsilon) && \text{// от деф. на } P \\
&\Leftrightarrow \alpha \in \mathcal{L}(P).
\end{aligned}$$

Доказателството на лемата следва до голяма степен [15, стр. 136].
Понеже граматиката е в нормална форма на Чомски, то десните страни на правилата имат дължина ≤ 2, което удовлетворява изискванията за Δ функцията.

Пример 5.5. Нека разгледаме

$$G = \begin{cases} S \to AS \mid BS \mid \varepsilon \\ A \to aA \mid a \\ B \to Bb \mid b. \end{cases}$$

Ще построим стеков автомат $P = \langle Q, \boldsymbol{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle$, такъв че $\mathcal{L}(P) = \mathcal{L}(G)$.

- $\boldsymbol{\Sigma} \stackrel{\text{деф}}{=} \{a, b\}$;
- $\Gamma \stackrel{\text{деф}}{=} \{A, S, B, a, b, \sharp\}$;
- $\boldsymbol{Q} \stackrel{\text{деф}}{=} \{\boldsymbol{s}, q, \boldsymbol{f}\}$;

Дефинираме релацията на преходите, следвайки конструкцията от *Теорема 5.1*:

$\boldsymbol{\Delta}(\boldsymbol{s}, \varepsilon, \sharp) = \{(q, S\sharp)\}$;
$\boldsymbol{\Delta}(q, \varepsilon, S) = \{(q, AS), (q, BS), (q, \varepsilon)\}$;
$\boldsymbol{\Delta}(q, \varepsilon, A) = \{(q, aA), (q, a)\}$;
$\boldsymbol{\Delta}(q, \varepsilon, B) = \{(q, Bb), (q, b)\}$;
$\boldsymbol{\Delta}(q, a, a) = \{(q, \varepsilon)\}$;
$\boldsymbol{\Delta}(q, b, b) = \{(q, \varepsilon)\}$;
$\boldsymbol{\Delta}(q, \varepsilon, \sharp) = \{(\boldsymbol{f}, \varepsilon)\}$.

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

$$R(\ell) \overset{\text{деф}}{=} (\forall A \in \boldsymbol{V})(\forall \alpha \in \boldsymbol{\Sigma}^\star)[\, A \overset{\ell}{\lhd} \alpha \implies (p, \alpha, A) \vdash^\star (p, \varepsilon, \varepsilon)\,].$$

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:

$$\frac{\dfrac{\dfrac{A \to_G a}{\boldsymbol{\Delta}(p, \varepsilon, A) \ni (p, a)}}{(p, a, A) \vdash_P (p, a, a)} \qquad \dfrac{\boldsymbol{\Delta}(p, a, a) \ni (p, \varepsilon)}{(p, a, a) \vdash_P (p, \varepsilon, \varepsilon)}}{(p, a, A) \vdash_P^2 (p, \varepsilon, \varepsilon)}$$

Второ, 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:

$$\frac{\dfrac{\dfrac{A \to_G BC}{\boldsymbol{\Delta}(p, \varepsilon, A) \ni (p, BC)}}{(p, \alpha_1\alpha_2, A) \vdash_P (p, \alpha_1\alpha_2, BC)} \qquad \dfrac{\dfrac{B \overset{\ell_1}{\lhd} \alpha_1}{(p, \alpha_1, B) \vdash_P^\star (p, \varepsilon, \varepsilon)} \qquad \dfrac{C \overset{\ell_2}{\lhd} \alpha_2}{(p, \alpha_2, C) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(p, \alpha_1\alpha_2, BC) \vdash_P^\star (p, \varepsilon, \varepsilon)}}{(p, \underbrace{\alpha_1\alpha_2}_{\alpha}, A) \vdash_P^\star (p, \varepsilon, \varepsilon)}$$

Доказателството на (б) отново ще проведем с *пълна индукция* по броя на стъпките ℓ в изчислението на стековия автомат. За целта, да разгледаме свойството

$$R(\ell) \overset{\text{деф}}{=} (\forall A \in V)(\forall \alpha \in \boldsymbol{\Sigma}^\star)[\, (p, \alpha, A) \vdash^\ell (p, \varepsilon, \varepsilon) \implies A \overset{\star}{\lhd} \alpha\,].$$

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:

$$\frac{\dfrac{\boldsymbol{\Delta}(p, \varepsilon, A) \ni (p, BC)}{(p, \alpha, A) \vdash_P (p, \alpha, BC)} \qquad (p, \alpha, BC) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(p, \alpha, A) \vdash_P^\ell (p, \varepsilon, \varepsilon)}$$

Знаем от *Твърдение 5.4*, 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:

$$(p, \alpha_1, B) \vdash_P^{\ell_1} (p, \varepsilon, \varepsilon) \overset{\textbf{(И.П.)}}{\Longrightarrow} B \lhd \alpha_1$$
$$(p, \alpha_2, C) \vdash_P^{\ell_2} (p, \varepsilon, \varepsilon) \overset{\textbf{(И.П.)}}{\Longrightarrow} C \lhd \alpha_2.$$

Сега обединяваме всичко, което имаме и получаваме извода:

$$\dfrac{A \to_G BC \qquad \dfrac{(p, \alpha_1, B) \vdash_P^{\ell_1} (p, \varepsilon, \varepsilon)}{B \lhd \alpha_1} \qquad \dfrac{(p, \alpha_2, C) \vdash_P^{\ell_2} (p, \varepsilon, \varepsilon)}{C \lhd \alpha_2}}{A \lhd \alpha}$$

Остана да разгледаме случая, когато за някое $a \in \mathbf{\Sigma}$ имаме следното:

$$\dfrac{\dfrac{\mathbf{\Delta}(p, \varepsilon, A) \ni (p, a)}{(p, \alpha, A) \vdash_P (p, \alpha, a)} \qquad (p, \alpha, a) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(p, \alpha, A) \vdash_P^{\ell} (p, \varepsilon, \varepsilon)}$$

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 $\mathbf{\Delta}(p, \varepsilon, A) \ni (p, a)$, то $A \to_G a$ и тогава $A \lhd \alpha$. □

Лема 5.2. За всеки стеков автомат P, съществува безконтекстна граматика G, такава че $\mathcal{L}(P) = \mathcal{L}(G)$.

Доказателство. Нека е даден стековия автомат

$$P = \langle Q, \mathbf{\Sigma}, \Gamma, \sharp, \Delta, \boldsymbol{s}, \boldsymbol{f} \rangle.$$

Ще дефинираме безконтекстна граматика G, за която $\mathcal{L}(P) = \mathcal{L}(G)$. Променливите на граматиката са

$$V = \{[q, A, p] \,:\, q, p \in \boldsymbol{Q} \;\&\; A \in \Gamma\}.$$

Правилата на G са следните:

- Началната променлива е $S \stackrel{\text{деф}}{=} [\boldsymbol{s}, \sharp, \boldsymbol{f}]$;
- Нека имаме $(r, BC) \in \mathbf{\Delta}(q, x, A)$, където $x \in \mathbf{\Sigma}_\varepsilon$. Тогава за всеки две състояния q' и p добавяме правилата:

$$[q, A, p] \to_G x[r, B, q'][q', C, p].$$

- Нека имаме $(r, B) \in \mathbf{\Delta}(q, x, A)$, където $x \in \mathbf{\Sigma}_\varepsilon$. Тогава за всяко състояние $p \in \boldsymbol{Q}$ добавяме правилата:

$$[q, A, p] \to_G x[r, B, p].$$

- Нека имаме $(p, \varepsilon) \in \mathbf{\Delta}(q, x, A)$, където $x \in \mathbf{\Sigma}_\varepsilon$. 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 $\alpha \in \mathbf{\Sigma}^\star$, произволни състояния q и p, и произволен символ $A \in \Gamma$, е изпълнено, че:

$$[q, A, p] \lhd \alpha \text{ точно тогава, когато } (q, \alpha, A) \vdash_P^\star (p, \varepsilon, \varepsilon). \tag{5.9}$$

($\Rightarrow$) Да напомним, че $\mathbf{\Sigma}_\varepsilon \stackrel{\text{деф}}{=} \mathbf{\Sigma} \cup \{\varepsilon\}$. Да разгледаме свойството

$$R(\ell) \stackrel{\text{деф}}{=} (\forall p)(\forall q)(\forall A)(\forall \alpha)[\,(q, \alpha, A) \vdash^\ell (p, \varepsilon, \varepsilon) \implies [q, A, p] \lhd \alpha\,].$$

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 $(p, \varepsilon) \in \mathbf{\Delta}(q, x, A)$. Тогава е

Забележка. Преди да преминем към доказателството, нека да разгледаме конструкция на безконтекстна граматика G по даден ДКА $\mathcal{A}$.

- Променливите на G са $[q, p]$, за всеки $q, p \in Q_\mathcal{A}$.
- Имаме нужда и от начална променлива S в граматика G и правилата $S \to_G [\boldsymbol{s}, f]$, за всяко $f \in \boldsymbol{F}_\mathcal{A}$.
- Другите правила на граматиката G са следните: $[q, p] \to_G a[q', p]$, където $q' = \delta_\mathcal{A}(q, a)$.

За да се убедим, че $\mathcal{L}(\mathcal{A}) = \mathcal{L}(G)$ е достатъчно да докажем следната еквивалентност:

$$[q, p] \stackrel{\star}{\lhd}_G \alpha \;\Leftrightarrow\; \delta^\star_\mathcal{A}(q, \alpha) = p.$$

В доказателството на лемата, основната идея е подобна, макар и леко усложнена. Ще 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.

ясно, че $\ell = 1$, $\alpha = x$ и според конструкцията на граматиката G имаме правилото $[q, A, p] \to_G x$, откъдето веднага следва, че $[q, A, p] \overset{1}{\lhd} x$.

Нека $\alpha = x\beta$, където $x \in \mathbf{\Sigma}_\varepsilon$.

- Ако $\mathbf{\Delta}(q, x, A) \ni (r, B)$, то можем да разбием изчислението по следния начин:

$$\dfrac{\dfrac{\mathbf{\Delta}(q, x, A) \ni (r, B)}{(q, x\beta, A) \vdash_P (r, \beta, B)} \qquad (r, \beta, B) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(q, \underbrace{x\beta}_{\alpha}, A) \vdash_P^{\ell} (p, \varepsilon, \varepsilon)}$$

  Сега можем да приложим индукционното предположение и да получим извода:

$$\text{(деф.)}\ \dfrac{\dfrac{\mathbf{\Delta}(q, x, A) \ni (r, B)}{[q, A, p] \to_G x[r, B, p]} \qquad \dfrac{(r, \beta, B) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{[r, B, p] \lhd \beta}\ \textbf{(И.П.)}}{[q, A, p] \lhd \underbrace{x\beta}_{\alpha}}$$

- Ако $\mathbf{\Delta}(q, a, A) \ni (r, BC)$, то можем да разбием изчислението по следния начин:

$$\dfrac{\dfrac{\mathbf{\Delta}(q, a, A) \ni (r, BC)}{(q, a\beta, A) \vdash_P (r, \beta, BC)} \qquad (r, \beta, BC) \vdash_P^{\ell-1} (p, \varepsilon, \varepsilon)}{(q, \underbrace{a\beta}_{\alpha}, A) \vdash_P^{\ell} (p, \varepsilon, \varepsilon)}$$

  Според *Твърдение 5.4*, 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:

$$(r, \beta_1, B) \vdash_P^{\ell_1} (q', \varepsilon, \varepsilon) \overset{\textbf{(И.П.)}}{\Longrightarrow} [r, B, q'] \overset{\star}{\lhd} \beta_1$$
$$(q', \beta_2, C) \vdash_P^{\ell_2} (p, \varepsilon, \varepsilon) \overset{\textbf{(И.П.)}}{\Longrightarrow} [q', C, p] \overset{\star}{\lhd} \beta_2.$$

  Също така имаме, че $\mathbf{\Delta}(q, x, A) \ni (r, BC)$ и според дефиницията на стековия автомат, в граматиката имаме правилото

$$[q, A, p] \to_G x[r, B, q'][q', C, p],$$

  където $x \in \mathbf{\Sigma}_\varepsilon$. Обединявайки всичко, получаваме извода в граматиката:

$$\dfrac{\dfrac{\boldsymbol{\Delta}(q,x,A) \ni (r,BC)}{[q,A,p] \to_G x[r,B,q'][q',C,p]} \quad \dfrac{(r,\beta_1,B) \vdash_P^{\ell_1} (q',\varepsilon,\varepsilon)}{[r,B,q'] \overset{\star}{\lhd} \beta_1} \quad \dfrac{(q',\beta_2,C) \vdash_P^{\ell_2} (p,\varepsilon,\varepsilon)}{[q',C,p] \overset{\star}{\lhd} \beta_2}}{[q,A,p] \overset{\star}{\lhd} \underbrace{x\beta_1\beta_2}_{\alpha}}$$

Пример 5.6. Да разгледаме стековия автомат от Пример 5.4.

- Началната променлива в граматиката е $S = [q, \sharp, f]$.
- Понеже $\boldsymbol{\Delta}(q, \varepsilon, \sharp) = \{(f, \varepsilon)\}$, то имаме правилото $[q, \sharp, f] \to_G \varepsilon$.
- Понеже $\boldsymbol{\Delta}(q, a, \sharp) = \{(q, a\sharp)\}$, то имаме правилата

  $$[q, \sharp, p] \to_G a[q, a, r][r, \sharp, p],$$

  за произволни състояния $r, p \in \{q, f\}$.
- Понеже $\boldsymbol{\Delta}(q, a, a) = \{(q, aa)\}$, то имаме правилата

  $$[q, a, p] \to_G a[q, a, r][r, a, p],$$

  за произволни $r, p \in \{q, f\}$.
- Понеже $\boldsymbol{\Delta}(q, b, a) = \{(q, \varepsilon)\}$, то имаме $[q, a, q] \to_G b$.

($\Leftarrow$) За тази посока, да разгледаме свойството

$$R(\ell) \overset{\text{деф}}{=} (\forall q)(\forall p)(\forall A)(\forall \alpha)[\, [q,A,p] \overset{\ell}{\lhd} \alpha \implies (q,\alpha,A) \vdash^{\star} (p,\varepsilon,\varepsilon)\,].$$

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 $\boldsymbol{\Delta}(q,x,A) \ni (p,\varepsilon)$. Тогава е ясно, че $(q,x,A) \vdash_P (p,\varepsilon,\varepsilon)$.

- Второ, да приемем, че имаме следния извод:

  $$\dfrac{[q,A,p] \to_G x[r,B,p] \qquad x \in \boldsymbol{\Sigma}_{\varepsilon} \qquad [r,B,p] \overset{\ell-1}{\lhd} \beta}{[q,A,p] \overset{\ell}{\lhd} \underbrace{x\beta}_{\alpha}}$$

  Тук отново е възможно $x = \varepsilon$. Това не е проблем, защото правим индукция по дължината на извода, а не по дължината на думата α. 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:

  $$\dfrac{\dfrac{\dfrac{[q,A,p] \to_G x[r,B,p]}{\boldsymbol{\Delta}(q,x,A) \ni (r,B)}}{(q,x\beta,A) \vdash_P (r,\beta,B)} \quad \dfrac{[r,B,p] \overset{\ell-1}{\lhd} \beta}{(r,\beta,B) \vdash_P^{\star} (p,\varepsilon,\varepsilon)}\ \textbf{(И.П.)}}{(q,\underbrace{x\beta}_{\alpha},A) \vdash_P^{\star} (p,\varepsilon,\varepsilon)}\ (\text{деф. на } G)$$

- Трето, да разгледаме случая:

$$\frac{[q,A,p] \to_G x[r,B,q'][q',C,p] \qquad [r,B,q'] \overset{\ell_1}{\lhd} \beta_2 \qquad [q',C,p] \overset{\ell_2}{\lhd} \beta_2}{[q,A,p] \overset{\ell}{\lhd} \underbrace{x\beta_1\beta_2}_{\alpha}}$$

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 $(r,BC) \in \mathbf{\Delta}(q,x,A)$. Обединявайки всичко, което знаем, получаваме изчислението:

$$\dfrac{\dfrac{\dfrac{[q,A,p] \to_G x[r,B,q'][q',C,p]}{\mathbf{\Delta}(q,x,A) \ni (r,BC)}}{(q,x\beta_1\beta_2,A) \vdash_P (r,\beta_1\beta_2,BC)} \qquad \dfrac{\dfrac{[r,B,q'] \overset{\ell_1}{\lhd} \beta_1}{(r,\beta_1,B) \vdash_P^\star (q',\varepsilon,\varepsilon)} \qquad \dfrac{[q',C,p] \overset{\ell_2}{\lhd} \beta_2}{(q',\beta_2,C) \vdash_P^\star (p,\varepsilon,\varepsilon)}}{(r,\beta_1\beta_2,BC) \vdash_P^\star (p,\varepsilon,\varepsilon)}}{(q,\underbrace{x\beta_1\beta_2}_{\alpha},A) \vdash_P^\star (p,\varepsilon,\varepsilon)}$$

□

Предишните две леми ни дават следната еквивалентност.

Теорема 5.1. Класът на езиците, които се разпознават от недетерминирани стекови автомати съвпада с класа на безконтекстните езици.

Забележка. Лесно можем да съобразим, че недетерминираните крайни автомати са опростен модел на недетерминираните стекови автомати. Оттук директно следва, че всеки регулярен език е безконтекстен.

5.3 Сечение с регулярен език

От *Твърдение 4.6* 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. Понеже имаме различни представяния на безконтекстните и регулярните езици, ще разгледаме различни подходи към доказателството на тази теорема.

Теорема 5.2. Нека L_1 е безконтекстен език и L_2 е регулярен език. Тогава тяхното сечение $L_1 \cap L_2$ е безконтекстен език.

Тук адаптираме доказателството от [15, стр. 144].

Упътване. Нека имаме стеков автомат

$$P_1 = \langle Q_1, \Sigma, \Gamma, \#, s_1, \Delta_1, f_1 \rangle, \text{ където } \mathcal{L}(P_1) = L_1,$$

и детерминиран краен автомат

$$\mathcal{A}_2 = \langle \Sigma, Q_2, s_2, \delta_2, F_2 \rangle, \text{ където } \mathcal{L}(\mathcal{A}_2) = L_2.$$

Сравнете с конструкцията от *Твърдение 3.2*.

Ще определим нов стеков автомат $\mathcal{M} = \langle Q, \Sigma, \Gamma, \natural, \Delta, s, f \rangle$, където:

- $Q \stackrel{\text{деф}}{=} Q_1 \times Q_2$;
- $s \stackrel{\text{деф}}{=} \langle s_1, s_2 \rangle$;
- $F \stackrel{\text{деф}}{=} \{f_1\} \times F_2$;
- Функцията на преходите Δ е дефинирана както следва:
  - Ако $(r_1, \gamma) \in \Delta_1(q_1, a, Y)$ и $r_2 = \delta_2(q_2, a)$, то

    $$\Delta(\langle q_1, q_2 \rangle, a, Y) \ni (\langle r_1, r_2 \rangle, \gamma).$$

    Симулираме едновременно изчислението и на двата автомата.

  - Ако $(r_1, \gamma) \in \Delta_1(q_1, \varepsilon, Y)$ и всяко $q_2 \in Q_2$, 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 е изпълнено свойството:

$$(\langle q_1, q_2 \rangle, \alpha, \gamma) \vdash^\star_{\mathcal{M}} (\langle p_1, p_2 \rangle, \varepsilon, \varepsilon) \quad \Leftrightarrow \quad (q_1, \alpha, \gamma) \vdash^\star_P (p_1, \varepsilon, \varepsilon) \text{ и } \delta^\star_{\mathcal{A}}(q_2, \alpha) = p_2.$$

□

Упътване. Можем да построим безконтекстна граматика G' директно по без- [4, стр. 32]
контекстна граматика G в нормална форма на Чомски и ДКА $\mathcal{A}$. Правилата на G' са следните:

- $[q, A, p] \to_{G'} [q, C, r][r, B, p]$, ако $A \to_G BC$ и $q, r, p \in \boldsymbol{Q}_{\mathcal{A}}$;
- $[q, A, p] \to_{G'} a$, ако $A \to_G a$ и $\delta_{\mathcal{A}}(q, a) = p$;
- $S' \to_{G'} [\boldsymbol{s}, S, f]$, за всяко $f \in \boldsymbol{F}_{\mathcal{A}}$.

Докажете, че е изпълнено свойството:

$$[q, A, p] \lhd_{G'} \alpha \text{ точно тогава, когато } A \lhd_G \alpha \text{ и } \delta^{\star}_{\mathcal{A}}(q, \alpha) = p.$$

□

Упътване. Нека G_1 е безконтекстна граматика за L_1 в нормална форма на Чомски и G_2 е регулярна граматика за L_2. Тогава дефинираме безконтекстна граматика G_3, където:

- $[X, A, Y] \to_{G_3} [X, B, Z][Z, C, Y]$, където $A \to_{G_1} BC$ и X, Y, Z са произволни променливи в G_2;
- $[X, A, Y] \to_{G_3} a$, където $A \to_{G_1} a$ и $X \to_{G_2} aY$;
- Началната променлива на G_3 е променливата S;
- Имаме и правилата $S \to_{G_3} [S_2, S_1, X]$, където S_1 е началната променлива в G_1, S_2 е началната променлива в G_2 и $X \to_{G_2} \varepsilon$.

За да се убедим, че $\mathcal{L}(G_3) = L_1 \cap L_2$ е достатъчно да се убедим, че е изпълнено свойството:

$$[X, A, Y] \lhd_{G_3} \alpha \text{ точно тогава, когато } A \lhd_{G_1} \alpha \text{ и } X \lhd_{G_2} \alpha Y.$$

□

Теорема 5.2 е удобна, когато искаме да докажем, че даден език не е безкон-
текстен. С нейна помощ можем да сведем езика до друг, за който вече знаем, че не е безконтекстен.

Пример 5.7. Да разгледаме езика

$$L = \{\omega \in \{a, b, c\}^{\star} \,:\, |\omega|_a = |\omega|_b = |\omega|_c\}.$$

Да допуснем, че L е безконтекстен език. Тогава, според *Теорема 5.2*, езикът

$$L' = L \cap \mathcal{L}(\boldsymbol{a}^{\star}\boldsymbol{b}^{\star}\boldsymbol{c}^{\star})$$

също е безконтекстен. Но $L' = \{a^n b^n c^n \ : \ n \in \mathbb{N}\}$, за който знаем от *Пример 4.6*, че *не* е безконтекстен. Достигнахме до противоречие. Следователно, L не е безконтекстен език.

Глава 6

Машини на Тюринг

> Turing's 'Machines'. These machines are humans who calculate. [23, § 1096].

Както при крайните автомати, ще започнем с разглеждането на детерминираните машини на Тюринг. След това ще разгледаме по-общия модел на недетерминираните машини на Тюринг и ще видим, че те разпознават същия клас от езици.

Тук до голяма степен следваме [22, Глава 3]. Понятието за машина на Тюринг има много еквивалентни дефиниции.

6.1 Детерминирани машини на Тюринг

Детерминирана машина на Тюринг ще наричаме осморка от вида

$$\mathcal{M} = \langle Q, \mathbf{\Sigma}, \Gamma, \delta, \sqcup, \boldsymbol{s}, \boldsymbol{f}, \boldsymbol{r} \rangle,$$

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, $\sqcup \in \Gamma \setminus \Sigma$;
- $\boldsymbol{s} \in Q$ - начално състояние;
- $\boldsymbol{f} \in Q$ - приемащо състояние;

Тези две състояния ще наричаме заключителни

- $\boldsymbol{r} \in Q$ - отхвърлящо състояние, където $\boldsymbol{f} \neq \boldsymbol{r}$;
- $\delta : Q' \times \Gamma \to Q \times \Gamma \times \{\triangleleft, \triangleright, \square\}$ - тотална функция на преходите, където $Q' = Q \setminus \{\boldsymbol{f}, \boldsymbol{r}\}$.

Това означава, че веднъж достигнем ли заключително състояние, не можем да правим повече преходи. Тук следваме [22, 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 $\boldsymbol{s}$ 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

  $$(\alpha, q, x\beta) \in \Gamma^\star \times Q \times \Gamma^+,$$

  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 четящата глава на машината е поставена върху x. По-нататък ще бъде удобно да положим

  $$\texttt{Conf}_{\mathcal{M}} \stackrel{\text{деф}}{=} \Gamma^\star \times Q \times \Gamma^+.$$

- 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 ω за да може лесно да работим със случая, когато $\omega = \varepsilon$. Това същото като на езици от типа на C++, където думите се представят като масив от символи, завършващи с '0'.

- **Началната конфигурация** за входната дума $\omega \in \Sigma^\star$ представлява тройката

$$(\varepsilon, \boldsymbol{s}, \omega\sqcup).$$

  Удобно е да положим

$$\texttt{init}_{\mathcal{M}}(\omega) \stackrel{\text{деф}}{=} (\varepsilon, \boldsymbol{s}, \omega\sqcup).$$

- **Приемаща конфигурация** представлява тройка от вида

$$(\lambda, \boldsymbol{f}, \rho),$$

  за произволни $\lambda \in \Gamma^\star$ и $\rho \in \Gamma^+$. Удобно е да положим

$$\texttt{Accept}_{\mathcal{M}} \stackrel{\text{деф}}{=} \Gamma^\star \times \{\boldsymbol{f}\} \times \Gamma^+.$$

- **Отхвърляща конфигурация** представлява тройка от вида

$$(\lambda, \boldsymbol{r}, \rho),$$

  за произволни $\lambda \in \Gamma^\star$ и $\rho \in \Gamma^+$. Удобно е да положим

$$\texttt{Reject}_{\mathcal{M}} \stackrel{\text{деф}}{=} \Gamma^\star \times \{\boldsymbol{r}\} \times \Gamma^+.$$

- Една конфигурация ще наричаме **заключителна**, ако тя е или приемаща или отхвърляща. Удобно е да положим

$$\texttt{Halt}_{\mathcal{M}} \stackrel{\text{деф}}{=} \texttt{Accept}_{\mathcal{M}} \cup \texttt{Reject}_{\mathcal{M}}.$$

Както за автомати, удобно е да дефинираме бинарна релация $\vdash_{\mathcal{M}}$ над множеството $\texttt{Conf}_{\mathcal{M}}$, 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!

$$\frac{x, z \in \Gamma \qquad \lambda, \rho \in \Gamma^\star}{(\lambda, xz\rho) \vdash_{y,\triangleright} (\lambda y, z\rho)} \qquad \frac{x \in \Gamma \qquad \lambda \in \Gamma^\star}{(\lambda, x) \vdash_{y,\triangleright} (\lambda y, \sqcup)}$$

$$\frac{x, z \in \Gamma \qquad \lambda, \rho \in \Gamma^\star}{(\lambda z, x\rho) \vdash_{y,\triangleleft} (\lambda, zy\rho)} \qquad \frac{x \in \Gamma \qquad \rho \in \Gamma^\star}{(\varepsilon, x\rho) \vdash_{y,\triangleleft} (\varepsilon, \sqcup y\rho)}$$

$$\frac{x \in \Gamma \qquad \lambda, \rho \in \Gamma^\star}{(\lambda, x\rho) \vdash_{y,\square} (\lambda, y\rho)}$$

Фигура 6.1: Дефиниция на релацията $\vdash_{y,d}$.

Ако няма опасност да се заблудим за коя точно машина на Тюринг $\mathcal{M}$ говорим, то е възможно да пишем просто $\vdash$ вместо $\vdash_{\mathcal{M}}$.

Сега вече сме готови да дефинираме релацията $\vdash_{\mathcal{M}}$.

$$\frac{\delta(q, x) = (q', y, d) \qquad (\lambda, x\rho) \vdash_{y,d} (\lambda', \rho')}{(\lambda, q, x\rho) \vdash_{\mathcal{M}} (\lambda', q', \rho')}$$

Фигура 6.2: 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 ω, ако

Важно е да имаме $\sqcup$ след думата ω, защото е възможно ω да е празната дума.

$$(\exists \kappa \in \texttt{Accept}_{\mathcal{M}})[\, \texttt{init}_{\mathcal{M}}(\omega) \vdash^\star_{\mathcal{M}} \kappa \,].$$

- Машината на Тюринг $\mathcal{M}$ **отхвърля** думата ω, ако

$$(\exists \kappa \in \texttt{Reject}_{\mathcal{M}})[\, \texttt{init}_{\mathcal{M}}(\omega) \vdash^{\star}_{\mathcal{M}} \kappa \,].$$

- Машината на Тюринг $\mathcal{M}$ **не приема** думата ω, ако $\mathcal{M}$ отхвърля ω или $\mathcal{M}$ никога не завършва при начална конфигурация $\texttt{init}_{\mathcal{M}}(\omega)$.

- Една машина на Тюринг се нарича **разрешител**, ако при всеки вход достига до заключително състояние, т.е. достига до приемаща или до отхвърляща конфигурация. С други думи, $\mathcal{M}$ е разрешител, ако

На англ. такава машина на Тюринг се нарича **decider** [22, стр. 170]. Може такива машини на Тюринг да се наричат и тотални [13, стр. 213]. Да се внимава, че в Манев понятията са различни.

$$(\forall \omega \in \Sigma^{\star})(\exists \kappa \in \texttt{Halt}_{\mathcal{M}})[\, \texttt{init}_{\mathcal{M}}(\omega) \vdash^{\star}_{\mathcal{M}} \kappa \,].$$

- Езикът, който се **разпознава** от машината $\mathcal{M}$ е:

$$\mathcal{L}(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \omega \in \Sigma^{\star} \,:\, (\exists \kappa \in \texttt{Accept}_{\mathcal{M}})[\, \texttt{init}_{\mathcal{M}}(\omega) \vdash^{\star}_{\mathcal{M}} \kappa \,] \,\}.$$

- Езикът L се нарича **полуразрешим**, ако съществува машина на Тюринг $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$. Ако една дума $\omega \in L$, то след крайно много стъпки ще достигнем до състоянието $\boldsymbol{f}$. Ако $\omega \notin L$, то не е ясно какво се случва с изчислението на $\mathcal{M}$ върху ω. Възможно е да достигнем до състоянието $\boldsymbol{r}$, но може да попаднем в безкрайно изчисление.

На англ. **semidecidable language**. В литературата се използва и названието **рекурсивно номеруем език**.

- Един език L се нарича **разрешим**, ако за него съществува *разрешител* $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$. В този случай се казва, че $\mathcal{M}$ разрешава езика L.

На англ. **decidable language**. В литературата се използва и названието **рекурсивен език**.

Твърдение 6.1. Ако L е разрешим език над азбуката Σ, то $\Sigma^{\star} \setminus L$ също е разрешим език.

Упътване. Просто разменяме $\boldsymbol{f}$ и $\boldsymbol{r}$. □

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 6.1. Да разгледаме езика вече познатия език $L \stackrel{\text{деф}}{=} \{a^n b^n c^n \ : \ n \in \mathbb{N}\}$. Нека да видим защо този език е разрешим. Идеята на алгоритъма, който ще разглеждаме 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. 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.

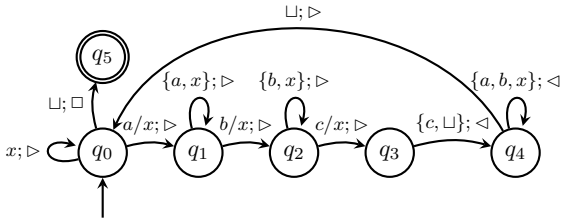
(4) Връща четящата глава в началото на лентата, т.е. чете наляво докато не срещне символа $\sqcup$. Връща се в стъпка (1).

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 $\boldsymbol{s} = q_0$ и $\boldsymbol{f} = q_5$;
- 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 $\boldsymbol{r}$.

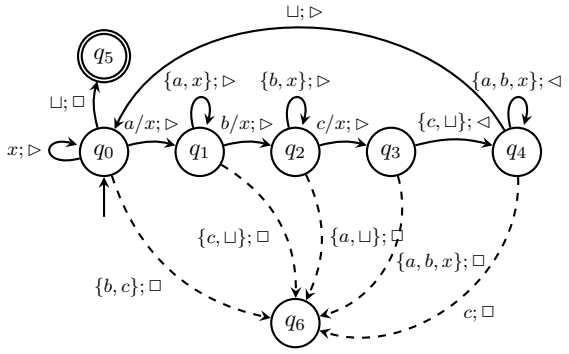


Фигура 6.4: Част от детерминистична машина на Тюринг $\mathcal{M}$, за която разрешава езика $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. Остава да се добави $\boldsymbol{r}$ и да се попълни δ функцията до тотална.

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 $q_6 = \boldsymbol{r}$ и за всяка двойка (q, z), за която функцията на преходите не е дефинирана, да сочи към $\boldsymbol{r}$. 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 $\boldsymbol{f}$ или $\boldsymbol{r}$. Заключаваме, че L е не само полуразрешим, но *разрешим* език.



Фигура 6.5: Детерминирана машина на Тюринг $\mathcal{M}$ разрешаваща $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. Отхвърлящото състояние е q_6, т.е. $\boldsymbol{r} \stackrel{\text{деф}}{=} q_6$.

Пример 6.2. Да разгледаме езика

$$L = \{\omega\sharp\omega \,:\, \omega \in \{a,b\}^\star\}.$$

Да напомним, че този език не е безконтекстен. В [10, стр. 155] е дадено по-различно решение. Тук следваме [22, стр. 173]. Там има малка грешка. 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.

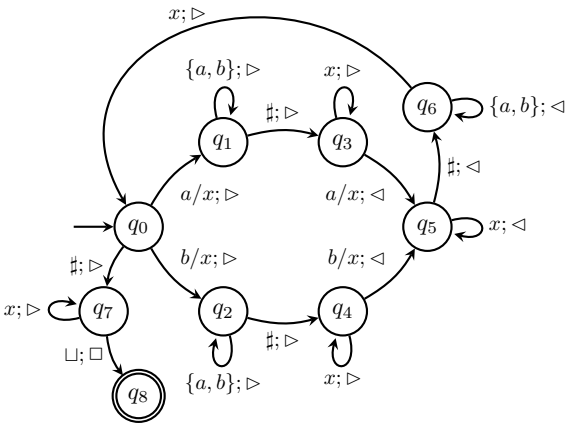
(4) Чете x-ове наляво по лентата докато не стигне $\sharp$.

(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 6.3, че ако разгледаме двулентова машина на Тюринг, то имаме алгоритъм със сложност $\mathcal{O}(n)$. Ще построим машина на Тюринг $\mathcal{M}$, за която $L = \mathcal{L}(\mathcal{M})$.

- $\Sigma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$ и $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp, x, \sqcup\}$;
- $Q \stackrel{\text{деф}}{=} \{q_0, q_1, \ldots, q_8\}$, като $\boldsymbol{s} \stackrel{\text{деф}}{=} q_0$ и $\boldsymbol{f} \stackrel{\text{деф}}{=} q_8$;



Фигура 6.6: Част от машина на Тюринг $\mathcal{M}$, която разрешава $\{\omega\sharp\omega \mid \omega \in \{a,b\}^\star\}$.

Да проследим изчислението на думата $ab\sharp ab$.

$$\begin{aligned}(q_0, \underline{a}b\sharp ab\sqcup) &\vdash (q_1, x\underline{b}\sharp ab\sqcup) \vdash (q_1, xb\underline{\sharp}ab\sqcup) \vdash (q_3, xb\sharp\underline{a}b\sqcup)\\ &\vdash (q_5, xb\underline{\sharp}xb\sqcup) \vdash (q_6, x\underline{b}\sharp xb\sqcup) \vdash (q_6, \underline{x}b\sharp xb\sqcup)\\ &\vdash (q_0, x\underline{b}\sharp xb\sqcup) \vdash (q_2, xx\underline{\sharp}xb\sqcup) \vdash (q_4, xx\sharp\underline{x}b\sqcup)\\ &\vdash (q_4, xx\sharp x\underline{b}\sqcup) \vdash (q_5, xx\underline{\sharp}x\underline{x}\sqcup) \vdash (q_5, xx\underline{\sharp}xx\sqcup)\\ &\vdash (q_6, x\underline{x}\sharp xx\sqcup) \vdash (q_0, xx\underline{\sharp}xx\sqcup) \vdash (q_7, xx\sharp\underline{x}x\sqcup)\\ &\vdash (q_7, xx\sharp x\underline{x}\sqcup) \vdash (q_7, xx\sharp xx\underline{\sqcup}) \vdash (q_8, xx\sharp xx\underline{\sqcup}).\end{aligned}$$

Може лесно да се съобрази, че тази машина на Тюринг може да се допълни до *тотална*.

6.2 Многолентови машини на Тюринг

Тук ще дефинираме един по-общ модел на машина на Тюринг, при който позволяваме да имаме няколко ленти, а не само една. Ще видим, че моделът с много на брой ленти има същата изразителна сила както и модела с една лента.

Многолентови детерминирани 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 $Q' = Q \setminus \{\boldsymbol{f}, \boldsymbol{r}\}$. Това означава, че имаме k на брой ленти, всяка с отделна четяща глава, която се движи независимо от другите.

Тук означаваме $\vec{a} = (a_0, \ldots, a_{k-1})$.

За две k-орки от думи $\vec{\alpha}$ и $\vec{\beta}$, дефинираме k-орката

$$\vec{\alpha} \cdot \vec{\beta} \stackrel{\text{деф}}{=} (\alpha_0 \cdot \beta_0, \alpha_1 \cdot \beta_1, \ldots, \alpha_{k-1} \cdot \beta_{k-1}).$$

В този случай е естествено да положим

$$\texttt{Conf}_{\mathcal{M}} \stackrel{\text{деф}}{=} (\Gamma^\star)^k \times Q \times (\Gamma^+)^k.$$

Сега дефинираме релацията $\vdash_{\mathcal{M}}$ над множеството $\texttt{Conf}_{\mathcal{M}}$ 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 6.7: Едностъпков преход в k-лентова детерминистична машина на Тюринг

Приемаме, че първоначално входната дума α е записана върху първата лента, а всички останали ленти са празни, т.е. запълнени са със символа $\sqcup$. Удобно да положим

$$\texttt{init}_{\mathcal{M}}(\omega) \stackrel{\text{деф}}{=} (\underbrace{\varepsilon, \ldots, \varepsilon}_{k}, \boldsymbol{s}, \underbrace{\omega\sqcup, \sqcup, \ldots, \sqcup}_{k}),$$

с което означаваме началната конфигурация при входна дума ω. Удобно е да положим също така и множествтото на приемащите конфигурации.

$$\texttt{Accept}_{\mathcal{M}} \stackrel{\text{деф}}{=} (\Gamma^\star)^k \times \{\boldsymbol{f}\} \times (\Gamma^+)^k.$$

Аналогично, можем да дефинираме и множеството на отхвърлящите конфигурации.

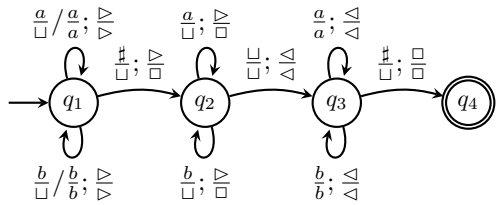
$$\texttt{Reject}_{\mathcal{M}} \stackrel{\text{деф}}{=} (\Gamma^\star)^k \times \{\boldsymbol{r}\} \times (\Gamma^+)^k.$$

Примери

Пример 6.3. Да видим как двулентова машина на Тюринг разрешава езика

$$L = \{\, \omega\sharp\omega \mid \omega \in \{a,b\}^\star \,\}.$$

- $Q = \{q_1, q_2, q_3, q_4, q_5\}$;
- $\boldsymbol{s} = q_1$, $\boldsymbol{f} = q_4$ и $\boldsymbol{r} = q_5$;
- $\Sigma \stackrel{\text{деф}}{=} \{a, b, \sharp\}$ и $\Gamma \stackrel{\text{деф}}{=} \{a, b, \sharp, \sqcup\}$;
- $\delta : Q' \times \Gamma^2 \to Q \times \Gamma^2 \times \{\triangleleft, \triangleright, \square\}^2$, където $Q' = \{q_1, q_2, q_3\}$.



Фигура 6.8: Непълно описание на двулентова детерминирана машина на Тюринг $\mathcal{M}$, за която $\mathcal{L}(\mathcal{M}) = L$. 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:

$$\begin{aligned}
(q_1, \tfrac{\hat{a}}{\hat{\sqcup}}\tfrac{b}{\sqcup}\tfrac{\sharp}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) &\vdash (q_1, \tfrac{a}{a}\tfrac{\hat{b}}{\hat{\sqcup}}\tfrac{\sharp}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_1, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\hat{\sharp}}{\hat{\sqcup}}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \\
&\vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{\hat{a}}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{\hat{a}}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \\
&\vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{a}{\sqcup}\tfrac{\hat{b}}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_2, \tfrac{a}{a}\tfrac{b}{b}\tfrac{\sharp}{\hat{\sqcup}}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\hat{\sqcup}}{\sqcup}) \\
&\vdash (q_3, \tfrac{a}{a}\tfrac{b}{\hat{b}}\tfrac{\sharp}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_3, \tfrac{a}{\hat{a}}\tfrac{b}{b}\tfrac{\sharp}{\sqcup}\tfrac{\hat{a}}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \\
&\vdash (q_3, \tfrac{\sqcup}{\hat{\sqcup}}\tfrac{a}{a}\tfrac{b}{b}\tfrac{\hat{\sharp}}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}) \vdash (q_4, \tfrac{\sqcup}{\hat{\sqcup}}\tfrac{a}{a}\tfrac{b}{b}\tfrac{\hat{\sharp}}{\sqcup}\tfrac{a}{\sqcup}\tfrac{b}{\sqcup}\tfrac{\sqcup}{\sqcup}).
\end{aligned}$$

Твърдение 6.2. За всяка k-лентова машина на Тюринг $\mathcal{M}$ съществува еднолентова машина на Тюринг $\mathcal{M}'$, такава че $\mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{M}')$.

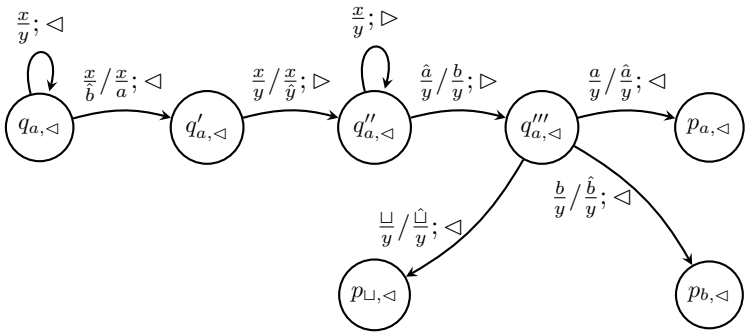
Упътване. Нека $\mathcal{M}$ е k-лентова машина на Тюринг. Ще построим еднолентова машина на Тюринг $\mathcal{M}'$, за която $\mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{M}')$. Да означим $\hat{\Gamma} = \{\hat{x} \mid x \in \Gamma\}$. Тогава азбуката на лентата на $\mathcal{M}'$ ще бъде $\Gamma' = \Sigma \cup (\hat{\Gamma} \cup \Gamma)^k$. Сега вместо да имаме k ленти със символи от Γ, ще имаме една лента със символи k-орки от $\hat{\Gamma} \cup \Gamma$. За да симулираме $\mathcal{M}$, използваме символите $\hat{x}$ за да маркират позицията на главите на $\mathcal{M}$, като във всяка компонента на лентата има точно по един символ от вида $\hat{x}$. При вход думата $\alpha = a_1 a_2 \cdots a_n$, която е поставена върху единствената лента на $\mathcal{M}'$, в случая на $k = 2$, първата работа на $\mathcal{M}'$ е да замени α с думата

$$\frac{\hat{a}_1}{\hat{\sqcup}} \frac{a_2}{\sqcup} \cdots \frac{a_n}{\sqcup}$$

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. □

В [22, 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].

Да видим по-подробно как можем да симулираме изчислението на двулентова машина на Тюринг $\mathcal{M}$ с еднолентовата машина на Тюринг $\mathcal{M}'$.



Фигура 6.9: 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

✍ Помислете как да обобщите тази идея за машина на Тюринг с n ленти. Важното е, че във всяко състояние кодираме крайна информация.

симулираната двулентова машина на Тюринг $\mathcal{M}$, главата на първата лента е върху символа a и главата на втората лента е разположена наляво/надясно/на същата позиция спрямо главата на първата лента.

- Във Фигура 6.9, $\frac{x}{y}$ 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.

6.3 Изчислими функции

Преди да дефинираме какво означава една функция да бъде изчислима с машина на Тюринг,

Когато искаме да дефинираме какво означава една функция да е изчислима с машина на Тюринг $\mathcal{M}$, ще разбираме, че резултатът от изчислението се запазва на последната лента на $\mathcal{M}$, като съдържанието ѝ е само това и четящата глава на последната лента е върху първата буква от резултата.

Удобно е да положим

Тази дефиниция работи и за $k = 1$, защото $A^0 = \{\varepsilon\}$.

$$\texttt{Accept}_{\mathcal{M}}(\omega) = (\Gamma^\star)^{k-1} \times \{\sqcup\}^\star \times \{\boldsymbol{f}\} \times (\Gamma^+)^{k-1} \times \{\omega\sqcup\}.$$

Една *тотална* функция $f : \Sigma^\star \to \Sigma^\star$ се нарича изчислима с машина на Тюринг $\mathcal{M}$, ако

$$(\forall \omega \in \Sigma^\star)(\exists \kappa \in \texttt{Accept}_{\mathcal{M}}(f(\omega)))[\, \texttt{init}_{\mathcal{M}}(\omega) \vdash^\star_{\mathcal{M}} \kappa \,].$$

Това означава, че машината на Тюринг $\mathcal{M}$ винаги завършва. Лесно се съобразява, че езикът $\texttt{Graph}(f) = \{\, \omega\sharp f(\omega) \mid \omega \in \Sigma^\star \,\}$ е разрешим.

Задача 6.1. Докажете, че съществуват функции от вида $f : \Sigma^\star \to \Sigma^\star$, които не са изчислими с машина на Тюринг.

Упътване. Всяка машина на Тюринг може да се кодира с естествено число. Това означава, че съществуват изброимо безкрайно много машини на Тюринг. От друга страна, съществуват неизброимо много функции от вида $f : \Sigma^\star \to \Sigma^\star$. □

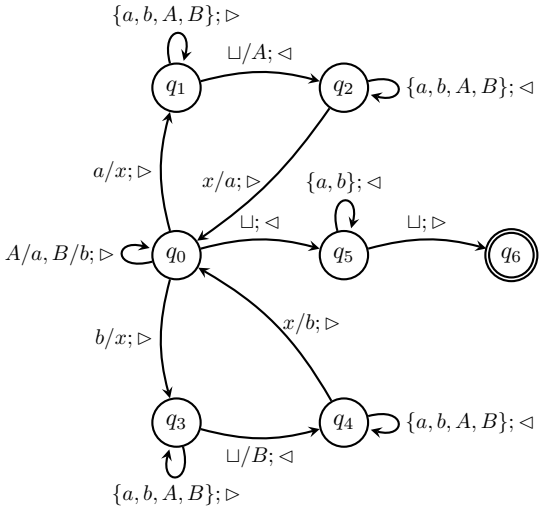
Пример 6.4. Да разгледаме тоталната функция

$$f : \{a,b\}^\star \to \{a,b\}^\star,$$

където $f(\omega) \stackrel{\text{деф}}{=} \omega \cdot \omega$.

Да видим защо f е изчислима с машина на Тюринг.

- $\Sigma \stackrel{\text{деф}}{=} \{a,b\}$ и $\Gamma \stackrel{\text{деф}}{=} \{a,b,x,A,B\}$;
- $\boldsymbol{s} \stackrel{\text{деф}}{=} q_0$ и $\boldsymbol{f} \stackrel{\text{деф}}{=} q_6$



Фигура 6.10: Машина на Тюринг за $f(\omega) = \omega \cdot \omega$.

Да проследим работата на $\mathcal{M}$ върху думата ab. Първо добавяме AB и така лентата съдържа $abAB$. След това заменяме A с a и B с b. Така най-накравя полуваме върху лентата думата $abab$.

$$\begin{aligned}(q_0, \underline{a}b\sqcup) &\vdash (q_1, x\underline{b}\sqcup) \vdash (q_1, xb\underline{\sqcup}) \vdash (q_2, x\underline{b}A) \vdash (q_2, \underline{x}bA)\\ &\vdash (q_0, a\underline{b}A) \vdash (q_3, ax\underline{A}) \vdash (q_3, axA\underline{\sqcup}) \vdash (q_4, ax\underline{A}B)\\ &\vdash (q_4, a\underline{x}AB) \vdash (q_0, ab\underline{A}B) \vdash (q_0, aba\underline{B}) \vdash (q_0, abab\underline{\sqcup})\\ &\vdash (q_5, aba\underline{b}) \vdash (q_5, ab\underline{a}b) \vdash (q_5, a\underline{b}ab)\\ &\vdash (q_5, \underline{a}bab) \vdash (q_5, \underline{\sqcup}abab) \vdash (q_6, \underline{a}bab).\end{aligned}$$

Не можем директно да започнем да копираме ω, защото така няма да знаем къде е края на първото копие на ω. Това можем да направим като първо запишем на лентата $\omega\sharp\omega$ и след това второто копие на ω го изместим с една позиция наляво.

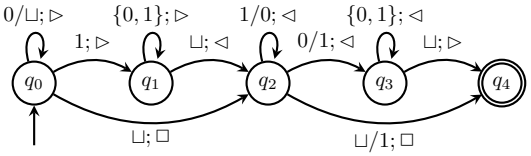
Забележка. Това пак става много по-лесно с двуленто-ва машина на Тюринг и сложността пак ще бъде $\mathcal{O}(n)$ вместо $\mathcal{O}(n^2)$.

Пример 6.5. Да разгледаме тоталната функция

$$f : \{0,1\}^\star \to 1 \cdot \{0,1\}^\star,$$

където $\overline{f(\omega)}_{(2)} \stackrel{\text{деф}}{=} \overline{\omega}_{(2)} + 1$. Нека да видим, че тази функция е изчислима с машина на Тюринг.

- $\Sigma \stackrel{\text{деф}}{=} \{0,1\}$ и $\Gamma \stackrel{\text{деф}}{=} \{0,1,\sqcup\}$;
- $\boldsymbol{s} = q_0$ и $\boldsymbol{f} \stackrel{\text{деф}}{=} q_4$.



Фигура 6.11: Машина на Тюринг изчисляваща f, за която $\overline{f(\omega)}_{(2)} = \overline{\omega}_{(2)} + 1$.

Да проследим изчислението на $\mathcal{M}$ върху вход 01011.

$$\begin{aligned}(q_0, \underline{0}1011\sqcup) &\vdash (q_0, \sqcup\underline{1}011\sqcup) \vdash (q_1, \sqcup1\underline{0}11\sqcup) \vdash (q_1, \sqcup10\underline{1}1\sqcup)\\ &\vdash (q_1, \sqcup101\underline{1}\sqcup) \vdash (q_1, \sqcup1011\underline{\sqcup}) \vdash (q_2, \sqcup101\underline{1}\sqcup)\\ &\vdash (q_2, \sqcup10\underline{1}0\sqcup) \vdash (q_2, \sqcup1\underline{0}00\sqcup) \vdash (q_3, \sqcup\underline{1}100\sqcup)\\ &\vdash (q_3, \underline{\sqcup}1100\sqcup) \vdash (q_4, \sqcup\underline{1}100\sqcup).\end{aligned}$$

Забележка. Изискваме $f(\omega)$ да започва с 1 за да може f да бъде функция, т.е. $f(\omega)$ е най-късият двоичен запис на числото $\overline{\omega}_{(2)} + 1$.

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:

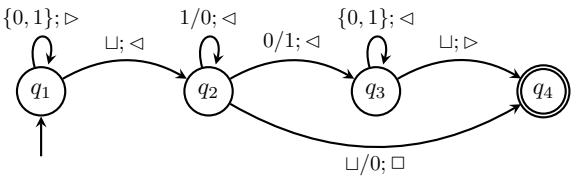
$$\varepsilon, 0, 1, \underbrace{00, 01, 10, 11}_{\text{от 0 до 3}}, \underbrace{000, 001, 010, 011, 100, 101, 110, 111}_{\text{от 0 до 7}}, 0000, 0001, \ldots$$

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 6.2. Нека $\Sigma = \{a_0, \ldots, a_{k-1}\}$. Да разгледаме функцията $f : \Sigma^\star \to \Sigma^\star$, за която $f(\alpha)$ е думата веднага след α в каноничната подредба на $\Sigma^\star$. Докажете, че f е изчислима с еднолентова детерминираната машина на Тюринг.

Упътване. Ако $\Sigma = \{0, 1\}$, то машината на Тюринг има следния вид:



Фигура 6.12: Генериране на следващата дума в каноничната наредба.

□

6.4 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 $Q' = Q \setminus \{\boldsymbol{f}, \boldsymbol{r}\}$.

Отново можем да дефинираме бинарна релация $\vdash_{\mathcal{N}}$ над множеството $\texttt{Conf}_{\mathcal{N}}$, 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 6.13: Преход в еднолентова недетерминирана машина на Тюринг $\mathcal{N}$

Тази дефиниция на релацията $\vdash^{\ell}_{\mathcal{N}}$ вече се повтаря няколко пъти.

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.

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

С $\vdash^{\star}_{\mathcal{N}}$ ще означаваме рефлексивното и транзитивно затваряне на релацията $\vdash_{\mathcal{N}}$ или с други думи,

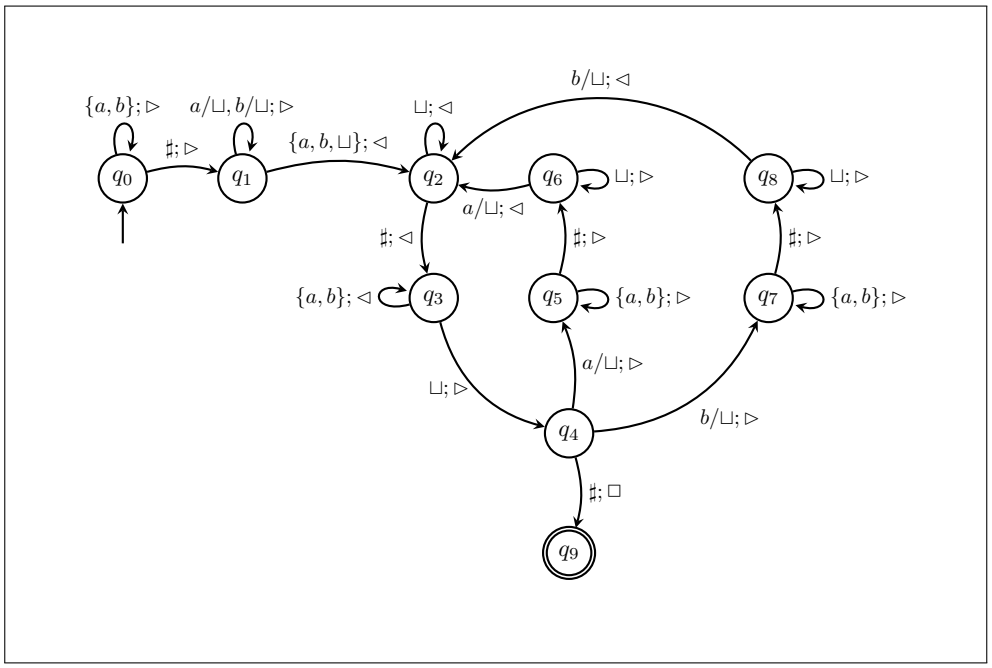
$$\kappa \vdash^{\star}_{\mathcal{N}} \kappa' \overset{\text{деф}}{\Leftrightarrow} (\exists \ell \in \mathbb{N})[\kappa \vdash^{\ell}_{\mathcal{N}} \kappa'].$$

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 $\boldsymbol{f}$. Възможно е много други изчисления при вход ω да завършват в $\boldsymbol{r}$ 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 $\boldsymbol{f}$ или $\boldsymbol{r}$.

Не е обяснено защо е разрешим.

Пример 6.6. 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.

$$
\begin{aligned}
(q_0, \underline{a}b\sharp aabb\sqcup) &\vdash (q_0, a\underline{b}\sharp aabb\sqcup) \vdash (q_0, ab\underline{\sharp}aabb\sqcup) \vdash (q_1, ab\sharp\underline{a}abb\sqcup) \\
&\vdash (q_1, ab\sharp \sqcup \underline{a}bb\sqcup) \vdash (q_2, ab\sharp\underline{\sqcup}abb\sqcup) \vdash (q_2, ab\underline{\sharp} \sqcup abb\sqcup) \\
&\vdash (q_3, a\underline{b}\sharp \sqcup abb\sqcup) \vdash (q_3, \underline{a}b\sharp \sqcup abb\sqcup) \vdash (q_3, \underline{\sqcup}ab\sharp \sqcup abb\sqcup) \\
&\vdash (q_4, \underline{a}b\sharp \sqcup abb\sqcup) \vdash (q_5, \sqcup\underline{b}\sharp \sqcup abb\sqcup) \vdash (q_5, \sqcup b\underline{\sharp} \sqcup abb\sqcup) \\
&\vdash (q_6, \sqcup b\sharp\underline{\sqcup}abb\sqcup) \vdash (q_6, \sqcup b\sharp \sqcup \underline{a}bb\sqcup) \vdash (q_2, \sqcup b\sharp\underline{\sqcup} \sqcup bb\sqcup) \\
&\vdash (q_2, \sqcup b\underline{\sharp} \sqcup \sqcup bb\sqcup) \vdash (q_3, \sqcup\underline{b}\sharp \sqcup \sqcup bb\sqcup) \vdash (q_3, \underline{\sqcup}b\sharp \sqcup \sqcup bb\sqcup) \\
&\vdash (q_4, \sqcup\underline{b}\sharp \sqcup \sqcup bb\sqcup) \vdash (q_7, \sqcup \sqcup \underline{\sharp} \sqcup \sqcup bb\sqcup) \\
&\vdash (q_8, \sqcup \sqcup \sharp\underline{\sqcup} \sqcup bb\sqcup) \vdash (q_8, \sqcup \sqcup \sharp \sqcup \underline{\sqcup}bb\sqcup) \\
&\vdash (q_8, \sqcup \sqcup \sharp \sqcup \sqcup\underline{b}b\sqcup) \vdash (q_2, \sqcup \sqcup \sharp \sqcup \underline{\sqcup} \sqcup b\sqcup) \\
&\vdash \cdots \vdash (q_4, \sqcup \sqcup \underline{\sharp} \sqcup \sqcup \sqcup b\sqcup) \vdash (q_9, \sqcup \sqcup \underline{\sharp} \sqcup \sqcup \sqcup b\sqcup)
\end{aligned}
$$

Теорема 6.1. Ако един език L се разпознава от *недетерминирана* машина на Тюринг, то L е разпознава и от *детерминирана* машина на Тюринг.

Доказателство. Нека имаме недетерминирана машина на Тюринг $\mathcal{N}$, за която $L = \mathcal{L}(\mathcal{N})$. Една дума α принадлежи на $\mathcal{L}(\mathcal{N})$ точно тогава, когато съществува изчисление, което започва с думата α върху лентата и след краен брой стъпки, следвайки функцията на преходите $\Delta_{\mathcal{N}}$, достига до състоянието $\boldsymbol{f}$. Сложността идва от факта, че за думата α може да имаме много различни изчисления, като само някои от тях завършват в $\boldsymbol{f}$. Ще построим детерминирана машина

В [10, стр. 164] не е добре обяснено.

на Тюринг $\mathcal{D}$, която последователно ще симулира всички възможни *крайни* изчисления за думата α, докато намери такова, което завършва в състоянието $\boldsymbol{f}$.

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 $\boldsymbol{f}$

Лесно се съобразява, че всяко изчисление на $\mathcal{N}$ може да се представи като крайна редица от елементи на $Q\times\Gamma\times\{\lhd,\rhd,\square\}$. 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:

$$\begin{aligned}
&(q_0, a, \square) \to 0,\ (q_0, a, \lhd) \to 1,\ (q_0, a, \rhd) \to 2,\\
&(q_0, b, \square) \to 3,\ (q_0, b, \lhd) \to 4,\ (q_0, b, \rhd) \to 5,\\
&(q_1, a, \square) \to 6, (q_1, a, \lhd) \to 7, (q_1, a, \rhd) \to 8,\\
&(q_1, b, \square) \to 9, (q_1, b, \lhd) \to 10,\ (q_1, b, \rhd) \to 11.
\end{aligned}$$

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

$$(\varepsilon, q_0, aba) \vdash_N (b, q_1, ba) \vdash_{\mathcal{N}} (b, q_1, aa) \vdash_{\mathcal{N}} (ba, q_0, a)$$

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 6.2* 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 $\boldsymbol{f}$ на $\mathcal{N}$, то машината $\mathcal{D}$ завършва успешно. В противен случай, на втората лента генерираме чрез функцията от *Задача 6.2* 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 6.1. 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* **??** 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 ($\boldsymbol{f}$ или $\boldsymbol{r}$).

- 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 $\boldsymbol{f}$.
- 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 $\boldsymbol{r}$. 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 $\mathcal{N}$.

□

Многолентови недетерминирани 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 $Q' = Q \setminus \{\boldsymbol{f}, \boldsymbol{r}\}$.

$$\frac{\Delta(q, \vec{x}) \ni (q', \vec{y}, \vec{d}) \qquad \forall i < k : (\lambda_i, x_i\rho_i) \vdash_{y_i, d_i} (\lambda_i', \rho_i')}{(\vec{\lambda}, q, \vec{x} \cdot \vec{\rho}) \vdash_{\mathcal{N}} (\vec{\lambda}', q', \vec{\rho}')}$$

Фигура 6.15: Едностъпков преход в k-лентова недетерминистична машина на Тюринг

Вече би трябвало да е ясно, че езиците, които се разпознават с k-лентови недетерминирани машини на Тюринг съвпадат с езиците, които се разпознават от еднолентови детерминирани машини на Тюринг.

6.5 Основни свойства

Твърдение 6.3. 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 6.7* 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 $\boldsymbol{f}$ и $\boldsymbol{r}$ състояния. Тогава $\mathcal{M}'$ също е разрешител и $\overline{L} = \mathcal{L}(\mathcal{M}')$. □

Твърдение 6.4. 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 6.5. 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 6.2 (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 $n \geq 2$;
- $q_i = \boldsymbol{s}$, $q_j = \boldsymbol{f}$ и $q_k = \boldsymbol{r}$.
- $\Sigma = \{x_1, \ldots, x_\ell\}$;
- $\Gamma = \{x_1, x_2, \ldots, x_s\}$, 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 всяка дума от вида

$$1110^\ell 110^i 110^j 110^k 11\ \texttt{code}_1\ 11\ \texttt{code}_2\ 11\ \cdots\ 11\ \texttt{code}_r\ 111$$

е код на машината на Тюринг $\mathcal{M}$.

По същия начин можем да дефинираме и код на краен автомат и стеков автомат.

Ако ω е код на машина на Тюринг, то с $\mathcal{M}_\omega$ ще означаваме машината на Тюринг, чийто код е ω. Ако ω не е код на машина на Тюринг, то с $\mathcal{M}_\omega$ ще означаваме машината на Тюринг, която има само две състояния $\boldsymbol{f}$ и $\boldsymbol{r}$, $\boldsymbol{s} = \boldsymbol{r}$. Това означава, че тази машина не прави никакви преходи и веднага завършва в състояние $\boldsymbol{r}$.

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 6.3. 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.

Твърдение 6.6. Съществуват следните функции:

- тотална изчислима функция f_1 на два аргумента, която има свойството:

$$\mathcal{L}(\mathcal{M}_{f_0(\omega,\rho)}) = \mathcal{L}(\mathcal{M}_\omega) \cap \mathcal{L}(\mathcal{M}_\rho).$$

- тотална изчислима функция f_2 на два аргумента, която има свойството:

$$\mathcal{L}(\mathcal{M}_{f_1(\omega,\rho)}) = \mathcal{L}(\mathcal{M}_\omega) \cup \mathcal{L}(\mathcal{M}_\rho).$$

- тотална изчислима функция f_3, която има свойството:

$$\mathcal{L}(\mathcal{M}_{f_3(\omega)}) = \{\alpha \in \{0,1\}^\star \mid \mathcal{M}_\omega \text{ не завършва върху } \omega \text{ за } \leq |\alpha| \text{ стъпки }\}.$$

- тотална изчислима функция f_4, която има свойството:

$$\mathcal{L}(\mathcal{M}_{f_4(\omega)}) = \{\alpha \in \{0,1\}^\star \mid \mathcal{M}_\omega \text{ завършва върху } \omega \text{ за } \leq |\alpha| \text{ стъпки }\}.$$

- тотална изчислима функция f_5, която има свойството:

$$\mathcal{L}(\mathcal{M}_{f_5(\omega)}) = \{\alpha \in \{0,1\}^\star \mid \omega \in \mathcal{L}(\mathcal{M}_\omega)\}.$$

Диагоналният език

Теорема 6.3. Диагоналният език

$$L_{\texttt{diag}} \stackrel{\text{деф}}{=} \{\, \omega \in \{0,1\}^\star \mid \omega \notin L(\mathcal{M}_\omega) \,\}$$

не се разпознава от машина на Тюринг, т.е. $L_{\texttt{diag}}$ **не** е полуразрешим език.

Това е версия на диагоналния метод на Кантор, с чиято помощ се доказва, че реалните числа са неизброимо много, т.е. има повече реални числа отколкото естествените.

Доказателство. Да допуснем, че $L_{\texttt{diag}}$ се разпознава от машина на Тюринг $\mathcal{M}$, т.е. $L_{\texttt{diag}} = \mathcal{L}(\mathcal{M})$. Тогава да видим какво имаме за думата $\ulcorner\mathcal{M}\urcorner$:

$$\ulcorner\mathcal{M}\urcorner \in L_{\texttt{diag}} \implies \ulcorner\mathcal{M}\urcorner \in \mathcal{L}(\mathcal{M}) \implies \ulcorner\mathcal{M}\urcorner \notin L_{\texttt{diag}},$$
$$\ulcorner\mathcal{M}\urcorner \notin L_{\texttt{diag}} \implies \ulcorner\mathcal{M}\urcorner \notin \mathcal{L}(\ulcorner\mathcal{M}\urcorner) \implies \ulcorner\mathcal{M}\urcorner \in L_{\texttt{diag}}.$$

Тук е добре една безкрайна таблица да се нарисува.

Достигаме до противоречие. □

Твърдение 6.7. Езикът

$$L_{\texttt{accept}} = \overline{L}_{\texttt{diag}} \stackrel{\text{деф}}{=} \{\, \omega \in \{0,1\}^\star \mid \omega \in \mathcal{L}(\mathcal{M}_\omega) \,\}$$

е полуразрешим, но не е разрешим.

Упътване. Лесно се съобразява, че $\overline{L}_{\texttt{diag}}$ е полуразрешим. Дефинираме машина на Тюринг $\mathcal{M}'$, която, при вход произволна дума ω, работи по следния начин:

Това можем да го направим, защото знаем, че $L_{\texttt{code}}$ е разрешим.

(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.

Получаваме, че за всяка дума ω е изпълнена еквивалентността

$$\omega \in \overline{L}_{\texttt{diag}} \Leftrightarrow \omega \in \mathcal{L}(\mathcal{M}'),$$

откъдето следва, че $\overline{L}_{\texttt{diag}}$ е полуразрешим език.

Ако допуснем, че $\overline{L}_{\texttt{diag}}$ е разрешим, то езикът $\overline{\overline{L}}_{\texttt{diag}} = L_{\texttt{diag}}$ би бил разрешим, което е противоречие, защото $L_{\texttt{diag}}$ не е е дори полуразрешим. □

Следствие 6.2. Класът на разрешимите езици строго се включва в класа на полуразрешимите езици.

Твърдение 6.8. Докажете, че езикът

$$L_{\texttt{halt}} = \{\omega \in \{0,1\}^\star \,:\, \mathcal{M}_\omega \text{ спира върху } \omega\}$$

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) $\mathcal{M}'$ симулира работата на $\mathcal{M}_{\texttt{halt}}$ върху ω.

(2) Ако симулацията завърши с резултат, че $\mathcal{M}_{\texttt{halt}}(\omega) = \boldsymbol{r}$, то $\mathcal{M}'(\omega) = \boldsymbol{r}$.

(3) Ако симулацията завърши с резултат, че $\mathcal{M}_{\texttt{halt}}(\omega) = \boldsymbol{f}$, то $\mathcal{M}'$ симулира работата на $\mathcal{M}_\omega$ върху ω, докато тя завърши.

Щом $\omega \in L_{\texttt{halt}}$, то знаем, че рано или късно тази симулация ще завърши в някое състояние.

- Ако $\mathcal{M}_\omega(\omega) = \boldsymbol{f}$, то $\mathcal{M}'(\omega) = \boldsymbol{f}$.
- Ако $\mathcal{M}_\omega(\omega) = \boldsymbol{r}$, то$\mathcal{M}'(\omega) = \boldsymbol{r}$.

□

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 6.4. Универсалният език

$$L_{\texttt{univ}} \stackrel{\text{деф}}{=} \{\, \omega\sharp\alpha \,:\, \alpha \in \mathcal{L}(\mathcal{M}_\omega) \,\}$$

е полуразрешим, но **не** е разрешим.

Разсъждението е много сходно с това защо $L_{\texttt{accept}}$ полуразрешим.
Ще наричаме $\mathcal{U}$ *универсална машина на Тюринг.*
✍ Съобразете, че езикът $L'_{\texttt{univ}} \stackrel{\text{деф}}{=} \{\omega\sharp\alpha \mid \alpha \notin \mathcal{L}(\mathcal{M}_\omega)\}$ **не** е полуразрешим.

Упътване. Първо да съобразим защо $L_{\texttt{univ}}$ е полуразрешим език. Дефинираме (многолентова) машина на Тюринг $\mathcal{U}$, която работи по следния начин:

- $\mathcal{U}$ проверява дали входната дума има вида $\omega\sharp\alpha$.
- Ако това е така, то то $\mathcal{U}$ симулира работата на $\mathcal{M}_\omega$ върху α.
  - Ако $\mathcal{M}_\omega$ завърши след краен брой стъпки като приеме α, то $\mathcal{U}$ приема $\alpha\sharp\omega$.
  - Ако $\mathcal{M}$ завърши след краен брой стъпки като отхвърли α, то $\mathcal{U}$ отхвърля $\alpha\sharp\omega$.
  - Ако $\mathcal{M}$ никога не завършва върху α, то очевидно $\mathcal{U}$ също никога не завършва върху $\omega\sharp\alpha$.
- Ако входната дума няма вида $\omega\sharp\alpha$, то $\mathcal{U}$ завършва веднага като отхвърля входната дума.

Получаваме, че

$$\gamma \in L_{\texttt{univ}} \;\Leftrightarrow\; \gamma \in \mathcal{L}(\mathcal{U}).$$

Сега да съобразим защо $L_{\texttt{univ}}$ не е разрешим език. За произволна дума ω имаме:

$$\omega \in \overline{L}_{\texttt{diag}} \;\Leftrightarrow\; \omega\sharp\omega \in L_{\texttt{univ}}.$$

Ако допуснем, че $L_{\texttt{univ}}$ е разрешим, то тогава $\overline{L}_{\texttt{diag}}$ е разрешим език, което е противоречие. □

6.6 Критерий за разрешимост

Сипсър нарича $\leq_m$ *mapping reducibility* [22, 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 6.4*, 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:

$$L_{\texttt{palin}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ съдържа само думи палиндроми}\}.$$

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 6.9. Езикът

$$L_{\Sigma^\star} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

не е разрешим.

Доказателство. Ще дефинираме тотална изчислима функция f, която при вход думата $\omega \in \{0,1\}^\star$, $f(\omega) \stackrel{\text{деф}}{=} \ulcorner\mathcal{M}'\urcorner$, където $\mathcal{M}'$, при вход произволна дума α, работи по следния начин:

$L_{\Sigma^\star}$ не е дори полуразрешим, но за момента не знаем как да докажем това.

(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 $\mathcal{M}_\omega(\omega) = \boldsymbol{f}$, то $\mathcal{M}'(\alpha) = \boldsymbol{f}$.

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 е изпълнено следното:

$$\omega \in L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \Sigma^\star \implies f(\omega) \in L_{\Sigma^\star},$$
$$\omega \notin L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \emptyset \implies f(\omega) \notin L_{\Sigma^\star},$$

което 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 6.3. Езикът

$$\overline{L}_{\texttt{empty}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \neq \emptyset\}$$

е полуразрешим, но не е разрешим.

Упътване. Съобразете, че в може да използвате функцията f от доказателството на *Твърдение 6.9* за да получите, че:

$$\omega \in L_{\texttt{accept}} \Leftrightarrow f(\omega) \in \overline{L}_{\texttt{empty}}.$$

□

Следствие 6.4. Езикът

$$L_{\texttt{empty}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

не е полуразрешим.

Упътване. Ако $L_{\texttt{empty}}$ беше разрешим, то $\overline{L}_{\texttt{empty}}$ 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 6.10. Езикът

$$L_{\texttt{reg}} \stackrel{\text{деф}}{=} \{\, \omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е регулярен език } \}$$

не е разрешим.

Доказателство. Да фиксираме един език, за който знаем, че не е регулярен, например, $\{0^n1^n \mid n \in \mathbb{N}\}$. Ще дефинираме тотална изчислима функция f, която при вход думата $\omega \in \{0,1\}^\star$, то $f(\omega) \stackrel{\text{деф}}{=} \ulcorner \mathcal{M}' \urcorner$, където $\mathcal{M}'$, при вход произволна дума α, работи така: [22, стр. 219]

(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 $\mathcal{M}_\omega(\omega) = \boldsymbol{f}$, то $\mathcal{M}'(\alpha) = \boldsymbol{f}$.

Получаваме, че:

Използваме наготово, че $\{0,1\}^\star$ е регулярен език.

$$\mathcal{L}(\mathcal{M}') = \begin{cases} \{0,1\}^\star, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ \{0^n1^n \,:\, n \in \mathbb{N}\}, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

Сега можем да заключим, че за произволна дума ω е изпълненото следното:

$$\omega \in L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \{0,1\}^\star \implies f(\omega) \in L_{\texttt{reg}},$$
$$\omega \notin L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \{0^n1^n \,:\, n \in \mathbb{N}\} \implies f(\omega) \notin L_{\texttt{reg}},$$

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 езици над фиксирана азбука Σ. Ще казваме,

че $\mathscr{S}$ е свойство на езиците над Σ, ако $\mathscr{S} \subseteq \mathcal{P}(\Sigma^\star)$. Например,

$$\mathscr{S} = \{L \subseteq \Sigma^\star \ : \ L \text{ е регулярен език}\}.$$

Ще представим това множество като език от кодовете на тези машини на Тюринг, т.е.

$$\texttt{Code}(\mathscr{S}) \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \ : \ \mathcal{L}(\mathcal{M}_\omega) \in \mathscr{S}\}.$$

$\mathscr{S}$ е **тривиално свойство**, ако $\texttt{Code}(\mathscr{S}) = \emptyset$ или $\texttt{Code}(\mathscr{S}) = \{0,1\}^\star$. Нека разгледаме изброимото множество от всички машини на Тюринг, които разпознават езиците от $\mathscr{S}$.

Можем да дефинираме и $\texttt{Code}(L)$, което е безкрайно изброимо множество, ако L е полуразрешим език.

Сега вече имаме достатъчно опит за да видим точно кои проблеми са разрешими.

[10, стр. 188]

Теорема 6.5 (Райс 1953 [19]). За всяко нетривиално свойство $\mathscr{S}$, $\texttt{Code}(\mathscr{S})$ е неразрешим.

Ако $\emptyset \in \mathscr{S}$, то правим горните разсъждения за класа от езици

$$\overline{\mathscr{S}} = \{L \subseteq \Sigma^\star \ : \ L \notin \mathscr{S}\}.$$

По аналогичен начин доказваме, че $\texttt{Code}(\overline{\mathscr{S}})$ не е разрешим език. Понеже

$$\texttt{Code}(\overline{\mathscr{S}}) = \{0,1\}^\star \setminus \texttt{Code}(\mathscr{S}),$$

то $\texttt{Code}(\mathscr{S})$ също не е разрешим език.

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 $f : \{0,1\}^\star \to \{0,1\}^\star$, дефинирана така, че за всяка дума ω, $f(\omega)$ е код на машина на Тюринг и

- при вход произволна дума α, $\mathcal{M}_{f(\omega)}$ работи така:
  (1) първоначално $\mathcal{M}_{f(\omega)}$ 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 $\mathcal{M}_\omega(\omega) = \boldsymbol{f}$, то $\mathcal{M}_{f(\omega)}$ започва да симулира работата на $\mathcal{M}_L$ върху входната дума α;
  (3) ако след краен брой стъпки симулацията завърши с резултат, че $\mathcal{M}_L(\alpha) = \boldsymbol{f}$, то $\mathcal{M}_{f(\omega)}(\omega) = \boldsymbol{f}$.

Така получаваме, че:

$$\mathcal{L}(\mathcal{M}_{f(\omega)}) = \begin{cases} L, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ \emptyset, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

Оттук заключаваме, че за произволна дума ω са изпълнено следното:

$$\omega \in L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L \implies f(\omega) \in \texttt{Code}(\mathscr{S}),$$
$$\omega \notin L_{\texttt{accept}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = \emptyset \implies f(\omega) \notin \texttt{Code}(\mathscr{S}),$$

което 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.

□

Нека сега да видим, че директно можем да получим много нови резултати, без нужда нищо да доказваме. Единствено трябва да съобразим, че всяко от тези свойства на полуразрешимите езици е нетривиално.

Следствие 6.5. За всяко от следните свойства $\mathscr{S}$ на полуразрешимите езици, $\texttt{Code}(\mathscr{S})$ **не** е разрешим език, където:

а) $\mathscr{S}$ е свойството празнота, т.е. езикът — $\mathscr{S} = \{\emptyset\}$.

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

не е разрешим;

б) $\mathscr{S}$ е свойството за пълнота, т.е. езикът — $\mathscr{S} = \{\Sigma^\star\}$.

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

не е разрешим;

в) $\mathscr{S}$ е свойството крайност, т.е. езикът — $\mathscr{S} = \{L \subseteq \Sigma^\star \,:\, |L| < \infty\}$.

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, |\mathcal{L}(\mathcal{M}_\omega)| < \infty\}$$

не е разрешим;

г) $\mathscr{S}$ е свойството безкрайност, т.е. езикът — $\mathscr{S} = \{L \subseteq \Sigma^\star \,:\, |L| = \infty\}$.

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, |\mathcal{L}(\mathcal{M}_\omega)| = \infty\}$$

не е разрешим;

д) $\mathscr{S}$ е свойството регулярност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е регулярен език }\}$$

не е разрешим;

е) $\mathscr{S}$ е свойството безконтекстност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е безконтекстен }\}$$

Това свойство е нетривиално, защото вече показахме, че $\{a^n b^n c^n \,:\, n \in \mathbb{N}\}$ е полуразрешим (дори разрешим) език, а знаем отдавна, че този език не е безконтекстен.

не е разрешим;

Тук също - вече сме разгледали примери за полуразрешими езици, които не са разрешими.

ж) $\mathscr{S}$ е свойството разрешимост, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е разрешим } \}$$

не е разрешим.

6.7 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 6.1. Нека $\mathscr{S}$ е свойство. Ако съществува безкраен полуразрешим език $L_0 \in \mathscr{S}$, който няма краен подезик в $\mathscr{S}$, то $\texttt{Code}(\mathscr{S})$ *не е* полуразрешим език.

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:

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 $\mathscr{S}$.

$$L_{\texttt{diag}} \leq_m \texttt{Code}(\mathscr{S}).$$

Ще дефинираме тотална изчислима функция $f : \{0,1\}^\star \to \{0,1\}^\star$, за която $f(\omega)$ е код на машина на Тюринг, за която, при вход произволна дума α, работи така:

(1) Първоначално $\mathcal{M}_{f(\omega)}$ 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 $\mathcal{M}_\omega(\omega) = \boldsymbol{f}$, то $\mathcal{M}_{f(\omega)}(\alpha) = \boldsymbol{r}$.

(3) В противен случай, т.е. $\mathcal{M}_\omega(\omega) = \boldsymbol{f}$ за повече от $|\alpha|$ на брой стъпки, то $\mathcal{M}_{f(\omega)}$ симулира работата на $\mathcal{M}_0$ върху α.

(4) Ако симулацията завърши с резултат, че $\mathcal{M}_0$ приема α, то $\mathcal{M}_{f(\omega)}$ завършва като приеме думата α.

Така получаваме, че:

$$\mathcal{L}(\mathcal{M}') = \begin{cases} \{\alpha \in L_0 \,:\, |\alpha| < \texttt{step}_\omega\}, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ L_0, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega), \end{cases}$$

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:

$$\omega \in L_{\texttt{diag}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L_0 \implies f(\omega) \in \texttt{Code}(\mathscr{S})$$
$$\omega \notin L_{\texttt{diag}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) \text{ е краен подезик на } L_0 \implies f(\omega) \notin \texttt{Code}(\mathscr{S})$$

които можем да обединим в следната еквивалентност:

$$\omega \in L_{\texttt{diag}} \Leftrightarrow f(\omega) \in \texttt{Code}(\mathscr{S}),$$

Оттук следва, че $\texttt{Code}(\mathscr{S})$ не е полуразрешим, защото от *Теорема 6.3* ние знаем, че $L_{\texttt{diag}}$ не е полуразрешим. □

✍ Защо не можем да използваме *Лема 6.1* за да докажем, че свойството празнота не е полуразрешимо, както и свойствата регулярност, разрешимост, полуразрешимост?

Следствие 6.6. Директно от *Лема 6.1* следва, че за всяко от следните свойства $\mathscr{S}$, $\texttt{Code}(\mathscr{S})$ **не** е полуразрешим език, където:

- $\mathscr{S}$ е свойството безкрайност, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, |\mathcal{L}(\mathcal{M}_\omega)| = \infty\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството за пълнота, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) = \Sigma^\star\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството неразрешимост, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ не е разрешим }\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството нерегулярност, т.е. езикът

  $$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ не е регулярен }\}$$

  не е полуразрешим.

Лема 6.2. Нека L_1 е език в $\mathscr{S}$ и нека L_2 е полуразрешим език, като $L_1 \subsetneqq L_2$ и $L_2 \notin \mathscr{S}$. Тогава $\texttt{Code}(\mathscr{S})$ не е полуразрешим език.

Това означава, че за полуразрешим език $\texttt{Code}(\mathscr{S})$, ако $L_1 \in \mathscr{S}$ и $L_1 \subseteq L_2$, като L_2 е полуразрешим, то $L_2 \in \mathscr{S}$.

Упътване. Нека $L_1 = \mathcal{L}(\mathcal{M}_1)$ и $L_2 = \mathcal{L}(\mathcal{M}_2)$. Ще докажем, че

$$L_{\texttt{diag}} \leq_m \texttt{Code}(\mathscr{S}).$$

Ще дефинираме тотална изчислима функция $f : \{0,1\}^\star \to \{0,1\}^\star$, за която $f(\omega)$ е код на машина на Тюринг, където:

(1) При вход дума α, $\mathcal{M}_{f(\omega)}$ 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 $\mathcal{M}_{f(\omega)}$ продължава като симулира работата $\mathcal{M}_2$ върху α.

(4) Ако симулацията на $\mathcal{M}_2$ завърши като приема думата α, то $\mathcal{M}_{f(\omega)}$ завършва като приема думата α.

Съобразете сами, че получаваме следното:

$$\mathcal{L}(\mathcal{M}_{f(\omega)}) = \begin{cases} L_2, & \text{ако } \omega \in \mathcal{L}(\mathcal{M}_\omega) \\ L_1, & \text{ако } \omega \notin \mathcal{L}(\mathcal{M}_\omega). \end{cases}$$

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:

$$\omega \in L_{\texttt{diag}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L_1 \implies f(\omega) \in \texttt{Code}(\mathscr{S})$$
$$\omega \notin L_{\texttt{diag}} \implies \mathcal{L}(\mathcal{M}_{f(\omega)}) = L_2 \implies f(\omega) \notin \texttt{Code}(\mathscr{S})$$

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 6.7. Директно от *Лема 6.2* следва, че за всяко от следните свойства $\mathscr{S}$, $\texttt{Code}(\mathscr{S})$ **не** е полуразрешим език, където:

- $\mathscr{S}$ е свойството крайност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, |\mathcal{L}(\mathcal{M}_\omega)| < \infty\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството празнота, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) = \emptyset\}$$

  не е полуразрешим;

- $\mathscr{S}$ е свойството разрешимост, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е разрешим}\}$$

не е полуразрешим;

- $\mathscr{S}$ е свойството регулярност, т.е. езикът

$$\texttt{Code}(\mathscr{S}) = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е регулярен}\}$$

не е полуразрешим.

Теорема 6.6 (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 6.1* 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 6.6*.

Лема 6.2 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 6.6*. □

6.8 Проблемът за съответствието на Пост

Пример за проблема за съответствието на Пост се нарича всяка крайна редица от елементи на $\Sigma^+ \times \Sigma^+$, които ние ще означаваме така:

На англ. *Post's correspondence problem* [9, стр. 392], но по-добре е обяснено в [22, стр. 227]. Сипсър нарича двойките домино.

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_n\\ \beta_n\end{bmatrix}.$$

Всяка една редица от този вид се нарича *пример* за PCP . Една непразна редица от индекси $i_1, i_2, \ldots, i_n$ се нарича *решение* на PCP примера, ако е изпълнено, че:

✍ Съобразете, че задачата е тривиална, ако:

- ако $|\alpha_i| = |\beta_i|$ за всяко i, или
- ако $|\Sigma| = 1$.

$$\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_n} = \beta_{i_1}\beta_{i_2}\cdots\beta_{i_n}.$$

Задача 6.4. Намерете решение на следния пример за PCP :

$$\begin{bmatrix}ab\\ abab\end{bmatrix}, \begin{bmatrix}b\\ a\end{bmatrix}, \begin{bmatrix}aba\\ b\end{bmatrix}, \begin{bmatrix}aa\\ a\end{bmatrix}.$$

Решение. [22, стр. 239].

$$\begin{bmatrix}ab\cdot ab\cdot aba\cdot b\cdot b\cdot aa\cdot aa\\ abab\cdot abab\cdot b\cdot a\cdot a\cdot a\cdot a\end{bmatrix}$$

□

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)

Казваме, че MPCP има решение, ако съществува произволна редица от индекси $i_1, \ldots, i_n$ (може и празна), такава че:

Тук искаме винаги да започваме с първата двойка.

$$\alpha_1\alpha_{i_1}\cdots\alpha_{i_n} = \beta_1\beta_{i_1}\cdots\beta_{i_n}.$$

Нека положим

$$\texttt{MPCP} \stackrel{\text{деф}}{=} \{\ulcorner P \urcorner \mid P \text{ е пример с решение за МПСП}\}.$$

Лема 6.3. 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:

$$\star\,\alpha = \star a_1 \star a_2 \cdots \star a_n$$
$$\alpha\star = a_1 \star a_2 \star \cdots a_n\star$$
$$\star\,\alpha\star = \star a_1 \star a_2 \star \cdots a_n \star .$$

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 6.8. 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 6.7 (Емил Пост [17])**.** 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 $\mathcal{M}_\omega$.

2) Започваме като добавяме за думата $\omega = a_1 \cdots a_n$ над азбуката Σ следната двойка $\begin{bmatrix} \sharp \\ \sharp q a_1 \cdots a_n \sqcup \sharp \end{bmatrix}$.

Горната част на доминото се опитва да настигне долната част.

3) Ако $\delta(q, a) = (p, b, \rhd)$, то добавяме двойката $\begin{bmatrix} qa \\ bp \end{bmatrix}$.

4) Ако $\delta(q, \sqcup) = (p, b, \rhd)$, то добавяме и двойката $\begin{bmatrix} q\sharp \\ bp\sharp \end{bmatrix}$.

5) Ако $\delta(q, a) = (p, b, \lhd)$, то добавяме двойките $\begin{bmatrix} xqa \\ pxb \end{bmatrix}$.

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.

6) Ако $\delta(q, \sqcup) = (p, b, \lhd)$, то добавяме двойките $\begin{bmatrix} xq\sharp \\ pxb\sharp \end{bmatrix}$.

7) Ако $\delta(q, a) = (p, b, \square)$, то добавяме двойката $\begin{bmatrix} qa \\ pb \end{bmatrix}$.

8) за всеки $x \in \Gamma$, добавяме $\begin{bmatrix} x \\ x \end{bmatrix}$. Освен това, добавяме и двойката $\begin{bmatrix} \sharp \\ \sharp \end{bmatrix}$.

9) За всеки $x \in \Gamma$, добавяме двойката $\begin{bmatrix} x\boldsymbol{f} \\ \boldsymbol{f} \end{bmatrix}$ и $\begin{bmatrix} \boldsymbol{f}x \\ \boldsymbol{f} \end{bmatrix}$.

Когато достигнем до приемащо състояние, то започваме да трием съдържанието на доминото за да можем да изравним двете части на доминото.

10) За да завършим, добавяме двойката $\begin{bmatrix} \boldsymbol{f}\sharp\sharp \\ \sharp \end{bmatrix}$.

□

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.

Следствие 6.9. Следният език

$$\texttt{AMBIG} \stackrel{\text{деф}}{=} \{\ulcorner G \urcorner \mid G \text{ е неоднозначна безконтекстна граматика}\}.$$

е полуразрешим, но не е разрешим език.

✍ Би трябвало да ви е очевидно защо AMBIG е полуразрешим език. Нали?

Упътване. Да разгледаме един пример за PCP над азбуката Σ:

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_n\\ \beta_n\end{bmatrix}.$$

По него можем ефективно да построим следната граматика:

$$G \begin{cases} S \to A \mid B \\ A \to \alpha_1 A c_1 \mid \alpha_2 A c_2 \mid \cdots \mid \alpha_n A c_n \mid \alpha_1 c_1 \mid \alpha_2 c_2 \mid \cdots \mid \alpha_n c_n \\ B \to \beta_1 B c_1 \mid \beta_2 B c_2 \mid \cdots \mid \beta_n B c_n \mid \beta_1 c_1 \mid \beta_2 c_2 \mid \cdots \mid \beta_n c_n, \end{cases}$$

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 PCP $\leq_m$ AMBIG. □

Следствие 6.10. 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 разрешим.

Ясно е, че INTERSECT е полуразрешим език, нали? Понеже INTERSECT не е разрешим, то е ясно, че допълнението на INTERSECT не е полуразрешим език. Това ще докажем повторно и в *Теорема 6.8*.

Упътване. Да разгледаме един пример за PCP над азбуката Σ:

$$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}, \begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}, \ldots, \begin{bmatrix}\alpha_n\\ \beta_n\end{bmatrix}.$$

По него можем ефективно да построим следните две безконтекстни граматики с правила:

$$G_1 \begin{cases} S_1 \to \alpha_1 S_1 c_1 \mid \alpha_2 S_1 c_2 \mid \cdots \mid \alpha_n S_1 c_n \mid \alpha_1 c_1 \mid \alpha_2 c_2 \mid \cdots \mid \alpha_n c_n \end{cases}$$

$$G_2 \begin{cases} S_2 \to \beta_1 S_2 c_1 \mid \beta_2 S_2 c_2 \mid \cdots \mid \beta_n S_2 c_n \mid \beta_1 c_1 \mid \beta_2 c_2 \mid \cdots \mid \beta_n c_n, \end{cases}$$

където $c_1, \ldots, c_n \notin \Sigma$. Така показахме, че PCP $\leq_m$ INTERSECT. □

6.9 Историята от всички изчисления

Да разгледаме машината на Тюринг $\mathcal{M}$. Една дума ω описва конфигурация на машина на Тюринг, ако $\omega \in \Gamma^\star Q \Gamma^+$. [10, стр. 201]

Твърдение 6.11. Да фиксираме една детерминистична машина на Тюринг $\mathcal{M}$. Тогава следните езици за безконтекстни:

$$\texttt{Valid}(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha\sharp\beta^{\texttt{rev}} \mid \alpha \in \Gamma^\star Q\Gamma^+ \;\&\; \beta \in \Gamma^+ Q\Gamma^\star \;\&\; \alpha \vdash_{\mathcal{M}} \beta \,\}$$

$$\texttt{Valid}'(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha^{\texttt{rev}}\sharp\beta \mid \alpha \in \Gamma^\star Q\Gamma^+ \;\&\; \beta \in \Gamma^+ Q\Gamma^\star \;\&\; \alpha \vdash_{\mathcal{M}} \beta \,\}$$

$$\texttt{Invalid}(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha\sharp\beta^{\texttt{rev}} \mid \alpha \in \Gamma^\star Q\Gamma^+ \;\&\; \beta \in \Gamma^+ Q\Gamma^\star \;\&\; \alpha \nvdash_{\mathcal{M}} \beta \,\}$$

$$\texttt{Invalid}'(\mathcal{M}) \stackrel{\text{деф}}{=} \{\, \alpha^{\texttt{rev}}\sharp\beta \mid \alpha \in \Gamma^\star Q\Gamma^+ \;\&\; \beta \in \Gamma^+ Q\Gamma^\star \;\&\; \alpha \nvdash_{\mathcal{M}} \beta \,\}.$$

Упътване. Да напомним първо как дефинираме релацията $\vdash_{\mathcal{M}}$:

$$\begin{aligned}
&(\lambda, q, xz\rho) \vdash_{\mathcal{M}} (\lambda y, p, z\rho) && \text{// ако } q \xrightarrow{x/y;\triangleright} p\\
&(\lambda z, q, x\alpha_2) \vdash_{\mathcal{M}} (\lambda, p, zy\rho) && \text{// ако } q \xrightarrow{x/y;\triangleright} p\\
&(\lambda, q, x) \vdash_{\mathcal{M}} (\lambda y, p, \sqcup) && \text{// ако } q \xrightarrow{x/y;\triangleleft} p\\
&(\varepsilon, q, x\alpha_2) \vdash_{\mathcal{M}} (\varepsilon, p, \sqcup y\rho) && \text{// ако } q \xrightarrow{x/y;\triangleleft} p\\
&(\lambda, q, x\alpha_2) \vdash_{\mathcal{M}} (\alpha_1, p, y\rho) && \text{// ако } q \xrightarrow{x/y;\square} p.
\end{aligned}$$

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.

$$\begin{aligned}
&\lambda q x z \rho\sharp\rho^{\texttt{rev}} z p y \lambda^{\texttt{rev}} && \text{// ако } q \xrightarrow{x/y;\triangleright} p\\
&\lambda z q x \rho\sharp\rho^{\texttt{rev}} y z p \lambda^{\texttt{rev}} && \text{// ако } q \xrightarrow{x/y;\triangleleft} p\\
&\lambda q x\sharp \sqcup p y \lambda^{\texttt{rev}} && \text{// ако } q \xrightarrow{x/y;\triangleleft} p\\
&q x \rho\sharp\rho^{\texttt{rev}} y \sqcup p && \text{// ако } q \xrightarrow{x/y;\triangleleft} p\\
&\lambda q x \rho\sharp\rho^{\texttt{rev}} y p \lambda^{\texttt{rev}} && \text{// ако } q \xrightarrow{x/y;\square} p
\end{aligned}$$

□

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 $\sqcup$.
- $\omega_1 \in \boldsymbol{s}\Sigma^\star$ описва начална конфигурация.
- $\omega_n \in \Gamma^\star \cdot \{\boldsymbol{f}\} \cdot \Gamma^\star$ описва приемаща конфигурация.
- За четно $i < n$, $\omega_i^{\mathtt{rev}} \vdash_{\mathcal{M}} \omega_{i+1}$;
- За нечетно $i < n$, $\omega_i \vdash_{\mathcal{M}} \omega_{i+1}^{\mathtt{rev}}$;

Езикът съставен от всички такива думи ще означаваме с $\mathtt{History}(\mathcal{M})$.

[10, стр. 201]

Лема 6.4. За всяка машина на Тюринг $\mathcal{M}$ е изпълнено, че:

$$\mathtt{History}(\mathcal{M}) = L_1 \cap L_2,$$

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:

$$L_1 \stackrel{\text{деф}}{=} (\mathtt{Valid}(\mathcal{M}) \cdot \{\sharp\})^\star \cdot (\{\varepsilon\} \cup \Gamma^\star \cdot \{\boldsymbol{f}\} \cdot \Gamma^\star \cdot \{\sharp\})$$
$$L_2 \stackrel{\text{деф}}{=} \{\boldsymbol{s}\} \cdot \Sigma^\star \cdot \{\sharp\} \cdot (\mathtt{Valid'}(\mathcal{M}) \cdot \{\sharp\})^\star \cdot (\{\varepsilon\} \cup \Gamma^\star \cdot \{\boldsymbol{f}\} \cdot \Gamma^\star \cdot \{\sharp\}),$$

□

Лема 6.5. Нека $\mathcal{M}$ е детерминирана машина на Тюринг, която за всеки вход прави поне две стъпки преди да спре. Тогава $\mathtt{History}(\mathcal{M})$ е безконтекстен (регулярен) точно тогава, когато $\mathcal{L}(\mathcal{M})$ е краен.

Упътване. Ясно е, че ако $\mathcal{L}(\mathcal{M})$ е краен, то $\mathtt{History}(\mathcal{M})$ 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 $\mathtt{History}(\mathcal{M})$ не е безконтекстен (регулярен). □

Задача 6.5. Обяснете как може ефективно да се кодира всяка безконтекстна граматика G като дума $\ulcorner G \urcorner$ над азбуката $\{0, 1\}$.

Теорема 6.8. Езикът

$$\overline{\texttt{INTERSECT}} = \{\ulcorner G_1 \urcorner \sharp \ulcorner G_2 \urcorner \,:\, \mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \emptyset\}$$

не е полуразрешим.

Упътване. Лесно се вижда, че $L_{\texttt{empty}} \leq_m \overline{\texttt{INTERSECT}}$. По дадена дума $\ulcorner \mathcal{M} \urcorner$, можем ефективно да намерим G_1 и G_2, за които $\mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \texttt{Accept}(\mathcal{M})$, т.е. съществува тотална изчислима функция f, за която

Това не ми трябва вече ? Да напомним, че от *Следствие 6.10* знаем, че INTERSECT не е разрешим, но е полуразрешим.

$$f(\ulcorner \mathcal{M} \urcorner) = \ulcorner G_1 \urcorner \cdot \ulcorner G_2 \urcorner.$$

Тогава ако L е полуразрешим език, то $L_{\texttt{empty}}$ е полуразрешим език, което е противоречие, защото

$$\ulcorner \mathcal{M} \urcorner \in L_{\texttt{empty}} \;\Leftrightarrow\; f(\ulcorner \mathcal{M} \urcorner) \in \overline{\texttt{INTERSECT}}.$$

□

Лема 6.6. За всяка машина на Тюринг $\mathcal{M}$, допълнението на $\texttt{History}(\mathcal{M})$, което означаваме като $\overline{\texttt{History}}(\mathcal{M})$, е безконтекстен език. Освен това, по кода на $\mathcal{M}$ можем ефективно да намерим код на безконтекстната граматика за $\overline{\texttt{History}}(\mathcal{M})$.

Упътване. Една дума α не е история на приемащо изчисление, ако е изпълнено някое от следните условия:

- α не е от вида $\omega_1 \sharp \omega_2 \sharp \cdots \sharp \omega_n \sharp$, където $\omega_i \in \Gamma^\star Q \Gamma^\star$, или

Можем да опишем това свойство с регулярен език

- 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:
  - $\omega_1 \notin \{\boldsymbol{s}\} \cdot \Gamma^\star$, или
  - $\omega_n \notin \Gamma^\star \cdot \{\boldsymbol{f}\} \cdot \Gamma^\star$, 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 6.9. За дадена азбука Σ, езикът

$$\texttt{All}_{\texttt{CFG}} \stackrel{\text{деф}}{=} \{\ulcorner G \urcorner \,:\, G \text{ е безконтекстна граматика и } \mathcal{L}(G) = \Sigma^\star\}$$

не е полуразрешим.

Упътване. Ще видим, че имаме следното свеждане:

$$L_{\texttt{empty}} \leq_m \texttt{All}_{\texttt{CFG}}.$$

Тук ще използваме, че ако $\mathcal{L}(\mathcal{M}) = \emptyset$, то $\overline{\texttt{History}}(\mathcal{M}) = \hat{\Sigma}^\star$, 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

$$f(\ulcorner \mathcal{M} \urcorner) = \ulcorner G \urcorner \text{ и } \mathcal{L}(G) = \overline{\texttt{History}}(\mathcal{M}).$$

Тогава, ако допуснем, че $\texttt{All}_{\texttt{CFG}}$ е полуразрешим език, то $L_{\texttt{empty}}$ е полуразрешим, защото

$$\ulcorner \mathcal{M} \urcorner \in L_{\texttt{Empty}} \;\Leftrightarrow\; f(\ulcorner \mathcal{M} \urcorner) \in \texttt{All}_{\texttt{CFG}}.$$

□

Следствие 6.11. 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.

[10, стр. 205]

По дадена дума ω можем ефективно да проверим дали тя кодира език от $\mathcal{C}$ или не.

Теорема 6.10 (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:

- ефективно затворен относно обединение;
- ефективно затворен относно конкатенация с регулярен език;
- "$= \Sigma^\star$"е неразрешим за достатъчно голяма Σ.

Нека P е нетривиално свойство на $\mathcal{C}$, което е изпълнено за всеки регулярен език и ако $L \in P$, то $a^{-1}(L) \in P$. Тогава езикът

Съществуват езици от $\mathcal{C}$, които не притежават свойството P и такива, които го притежават.

$$\ulcorner P \urcorner \stackrel{\text{деф}}{=} \{\ulcorner L \urcorner \,:\, L \in P\}$$

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

$$\hat{L} \stackrel{\text{деф}}{=} \Sigma^\star \cdot \{\sharp\} \cdot L_0 \cup \; L \cdot \{\sharp\} \cdot \Sigma^\star.$$

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:

$$L = \Sigma^\star \;\; \Leftrightarrow \;\; \hat{L} \in P. \tag{6.1}$$

- Ако $L = \Sigma^\star$, то $\hat{L}$ е регулярен, защото тогава $\hat{L} = \Sigma^\star \cdot \{\sharp\} \cdot \Sigma^\star$ е очевидно регулярен и от избора на P, $\hat{L} \in \mathcal{C}$.
- Ако $L \neq \Sigma^\star$, то нека да фиксираме дума $\omega \notin L$. Ако допуснем, че $\hat{L} \in P$, то езикът за езикът $(\omega\sharp)^{-1}(\hat{L}) = L_0$ също ще е изпълнено свойството P, което е противоречие с избора на L_0.

Ако $L \in P$, то за $\beta^{-1}(L) \in P$.

От (6.1) следва, че P е разрешимо свойство точно тогава, когато въпросът "$= \Sigma^\star$" за езиците от $\mathcal{C}$ е разрешим, което е противоречие. □

Следствие 6.12. Въпросът дали една безконтекстна граматика описва регулярен език е неразрешим. По-точно, езикът

$$\texttt{Reg} = \{\ulcorner G \urcorner \,:\, \mathcal{L}(G) \text{ е регулярен език}\}$$

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 6.9*, че $= \Sigma^\star$ за

безконтекстни граматики е неразрешим за достатъчно голяма азбука Σ. Тогава от теоремата на Грейбах следва, че $\texttt{Reg}$ е неразрешим език. $\square$

6.10 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 6.6. Докажете, че класът $\mathcal{P}$ е затворен относно допълнение, обединение, сечение и конкатенация.

Задача 6.7. Докажете, че класът $\mathcal{P}$ е затворен относно операцията звезда на Клини.

Теорема 6.11. $\mathcal{NP} \subseteq \mathcal{EXP}$.

Твърдение 6.12. За азбука Σ от поне две букви, можем да обобщим някои от резултатите от предишните глави:

$$\texttt{REG} \subsetneq \texttt{CFG} \subsetneq \mathcal{P}.$$

Упътване. Езикът $\{a^n b^n c^n \mid n \in \mathbb{N}\} \in \mathcal{P}$, но не е безконтекстен. □

6.11 Задачи

Задача 6.8. Докажете директно, без да се позововавате на теоремата на Райс, че следните езици не са разрешими:

- $[L_{\texttt{Dec}} = \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega) \text{ е разрешим}\}$.
- $L_{\texttt{palin}} \stackrel{\text{деф}}{=} \{\omega \in \{0,1\}^\star \,:\, \mathcal{L}(\mathcal{M}_\omega)$ съдържа само думи палиндроми$\}$.

Задача 6.9. 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 6.10. 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 6.11. Докажете, че езикът

$$L = \{\omega\sharp\alpha \mid \mathcal{M}_\omega \text{прави движение наляво при работата си върху вход } \alpha\}$$

е разрешим.

Упътване. Нужно е да симулирате работата на $\mathcal{M}$ върху ω само за $|\omega|+|Q^{\mathcal{M}}|+1$ на брой стъпки. □

Задача 6.12. 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.

Азбучен указател

$R^\star$, 14
$\mathcal{L}_G(X)$, 130
$\mathbf{\Delta}^\star$, 79
$\boldsymbol{\delta}^\star$, 53
$\texttt{Code}(L)$, 226
$\texttt{Code}(\mathscr{S})$, 226
ε-правила, 152

Ардън, 28
Бжозовски, 62, 77
Винер, 9
Грейбах, 240
ДКА, 52
Де Морган, 9
Кантор, 10
Клини, 23, 89
Клини-Пост, 216
Куратовски, 9
Пост, 233, 234
Рабин, 79
Райс, 226, 232
Скот, 79
Тюринг, 197
Чомски, 155
Шапиро, 232
автомат
    детерминиран, 52
    недетерминиран (НКА), 79
    недетерминиран стеков, 179
автомат на Бжозовски, 62
автоматни езици
    допълнение, 60
    обединение, 60
    сечение, 59
азбука, 21
апроксимация на безконтекстен език, 130
безконтекстен език, 124
буква, 21
декартово произведение, 10
динамично програмиране, 159
дума, 21
    конкатенация, 23
    обръщане, 23
    отрез, 24
    празна, 21
дълго правило, 154
език, 22
    автоматен, 53
    детерминистично експоненциално разрешим, 243
    детерминистично полиномиално разрешим, 243
    звезда на Клини, 23
    недетерминистично полиномиално разрешим, 243
    неполуразрешим, 220
    неразрешим, 222
    полуразрешим, 201, 220
    разрешим, 201
    регулярен, 25
изоморфизъм, 71
индукция, 17
история на приемащо изчисление, 238
каноничен автомат, 62
конфигурация, 179
лема за покачването
    безконтекстни езици, 144

регулярни езици, 98
машина на Тюринг
детерминирана, 197
детерминистично полиномиално ограничена, 243
заключителна конфигурация, 199
конфигурация, 198
многолентова, 204
моментно описание, 198
начална конфигурация, 199
недетерминирана, 212
недетерминистично полиномиално ограничена, 243
отхвърляща конфигурация, 199
полиномиално ограничена, 243
приемаща конфигурация, 199
разрешител, 201
минимален автомат, 77
минимизация, 77
множества, 7
обединение, 7
разлика, 8
сечение, 7
степенно множество, 8
моментно описание, 179
наредба
канонична, 211
лексикографска, 24
наредена двойка, 9
недетерминизъм, 79
нормална форма на Чомски, 155
образ, 46
преименуващи правила, 153
префикс, 24
проблем за съответствието, 233
пълна индукция, 19

регулярен израз, 25
регулярни операции
звезда на Клини, 26
конкатенация, 25
обединение, 25
релация
антисиметрична, 12
композиция, 14
рефлексивна, 12
рефлексивно затваряне, 14
симетрична, 12
транзитивна, 12
транзитивно затваряне, 14
суфикс, 24
функция
биекция, 10
инекция, 10
сюрекция, 10
хомоморфизъм, 46

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] Christos Papadimitriou и Harry Lewis. *Elements of the Theory of Computation*. Prentice-Hall, 1998.

[16] 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.

[17] Emil L. Post. “A variant of a recursively unsolvable problem”. В: *Bull. Amer. Math. Soc.* 52 (1946), с. 264—268. doi: 10.1090/S0002-9904-1946-08555-9.

[18] 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.

[19] 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.

[20] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009. doi: 10.1017/CBO9781139195218.

[21] Jeffrey Shallit. *A Second Course in Formal Languages and Automata Theory*. Cambridge University Press, 2008.

[22] Michael Sipser. *Introduction to the Theory of Computation*. 3-е изд. Cengage Learning, 2012.

[23] 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.

[24] Ludwig Wittgenstein. *Tractatus Logico-Philosophicus*. Routledge, 1974.