

# Дискретна математика

доц. д-р Тодорка Глушкова,  
Катедра „Компютърни технологии“, ФМИ

7



# **АЗБУКИ, ДУМИ И ФОРМАЛНИ ЕЗИЦИ**

# Въведение

Теоретичната информатика се занимава с:

- формални езици,
- теория на автоматите,
- логика,
- разработка и анализ на алгоритми,
- формална семантика и дава основите за компилатори и математическото формализиране на проблеми.

Тя е формалният „гръбнак“ на информатиката. Формални системи, автомати, графи и синтактични диаграми се използват за точно описание на вътрешната логика на формални проблеми. Обикновено тази формална стъпка е основна част от решението на същинския проблем.

# Формални езици

- В математиката, логиката и компютърните науки, **формален език** е това множество от думи с крайна дължина (тоест буквени низове) извлечено от дадена крайна азбука.
- Научната теория, за която формалните езици са обект на изучаване се нарича *теория на формалните езици*.

# Формални езици

**Например:** *Азбука може да бъде  $\{c,d\}$  и низ към тази азбука може да бъде  $cddddc$ . Типичен език на тази азбука, съдържащ стринга  $cddddc$ , ще бъде множеството от всички стрингове, които съдържат същият брой  $c$  и  $d$  символи.*

- Празната дума (низ с нулева дължина) е разрешен и често означаван като  $\epsilon$ ,  $\varepsilon$  или  $\Lambda$ . Докато азбуката е крайно множество и всеки стринг има крайна дължина, то езикът може съвсем спокойно да се състои от безкрайно много стрингове.

# Формални езици

- Дефиниция : **Азбука** (или речник) се нарича всяко крайно множество от символи. **Символите** се наричат букви. Азбуките бележим с главните латински и гръцки букви K,V,W,Σ и др. Символите ще бележим с малките латински букви.
- Дефиниция: Всяка крайна редица от символи на дадена азбука ще наричаме **дума** над тази азбука. Думите ще бележим с малките гръцки букви:  
 $\alpha, \beta, \chi \dots$

# Формални езици

- Броят на символите в думата определя нейната **дължина**. Думата с дължина 0 е празната дума  $\varepsilon$ .
- Дефиниция: Две думи са **равни**, когато имат една и съща дължина и еднакви първи, втори и т.н. букви, т.е.  $\alpha = a_1a_2\dots a_n$ ;  $\beta = b_1b_2\dots b_n$ .
- Тогава  $\alpha = \beta \Leftrightarrow a_i = b_i$  за всяко  $i = 1\dots n$ .

# Формални езици

- Дефиниция: **Формален език** над дадена азбука ще наричаме всяко множество от думи над тази азбука. Бележим с  $L$ .
- Пример: Нека  $V=\{0,1\}$  и  $W=\{if, then, else, for, do, a, b, c\}$  са две азбуки. Тогава  $\varepsilon$ ,  $110101$  и  $00$  са думи над  $V$  с дължина  $0, 6, 2$ .
- *If b then for c do if b then a else c, if b then a else c* - са думи над  $W$  с дължина  $12$  и  $6$ .



# Други примери за формални езици

- множество на всички думи от  $\{a,b\}$
- множество  $\{ a^n : n \text{ е естествено число по-голямо от единица} \}$  (където  $a^n$  означава  $a$  повторено  $n$  пъти)
- множество от синтактично правилни програми за даден програмен език.

# Операции над думи и формални езици

Дефиниция: **Конкатинация** (съединение) на думите  $\alpha$  и  $\beta$  ще наричаме непосредственото записване на  $\beta$  след  $\alpha$  и ще означаваме  $\alpha\beta$ . Ако  $\alpha = a_1 \dots a_n$ ;  $\beta = b_1 \dots b_n$ , то  $\alpha\beta = a_1 \dots a_n b_1 \dots b_n$ .

Очевидно получената дума  $\alpha\beta$  е също дума над разглежданата азбука, следователно конкатенацията е вътрешна операция.

- Конкатенацията не е симетрична операция  
( $\alpha = 00$ ;  $\beta = 11 \Rightarrow \alpha\beta = 0011$ , а  $\beta\alpha = 1100$ )
- За всеки три думи  $\alpha, \beta, \chi$  е изпълнено:  $(\alpha\beta)\chi = \alpha(\beta\chi)$ .  
Следователно тя е асоциативна операция.
- Пример: (аз имам)компютър = аз(имам компютър)

# Операции над думи и формални езици

- Дефиниция: Нека  $\alpha$  е дума. Определяме **степените** на  $\alpha$  така:

$$\alpha^0 = \varepsilon; \alpha^1 = \alpha; \alpha^2 = \alpha\alpha \text{ и т.н.}$$

- Нека  $L1$  и  $L2$  са формални езици. Тогава:
  - а) обединението на езиците  $L1 \cup L2 = \{w: w \in L1 \vee w \in L2\}$ , където  $w$  е дума
  - б) сечение на езиците  $L1 \cap L2 = \{w: w \in L1 \wedge w \in L2\}$ ;
  - в) разлика на езици  $L1 - L2 = \{w: w \in L1 \wedge w \notin L2\}$ ;
  - г) произведение на езици  $L1.L2 = \{\alpha\beta: \alpha \in L1, \beta \in L2\} \Rightarrow$  от всяка конкатенация на думи от  $L1$  и  $L2$ .

# Примери

Пример:  $L1 = \{0,01\}$ ;  $L2 = \{1,11\}$

$L1 \cup L2 = \{0,01,1,11\}$

$L1 \cap L2 = \{\emptyset\}$

$L1 - L2 = \{0,01\}$

$L1.L2 = \{01,011,0111\}$

Забележка: От факта, че конкатенацията е асоциативна, но не е комутативна следва, че произведението на езици е асоциативна, но не комутативна операция.

# Операции над думи и формални езици

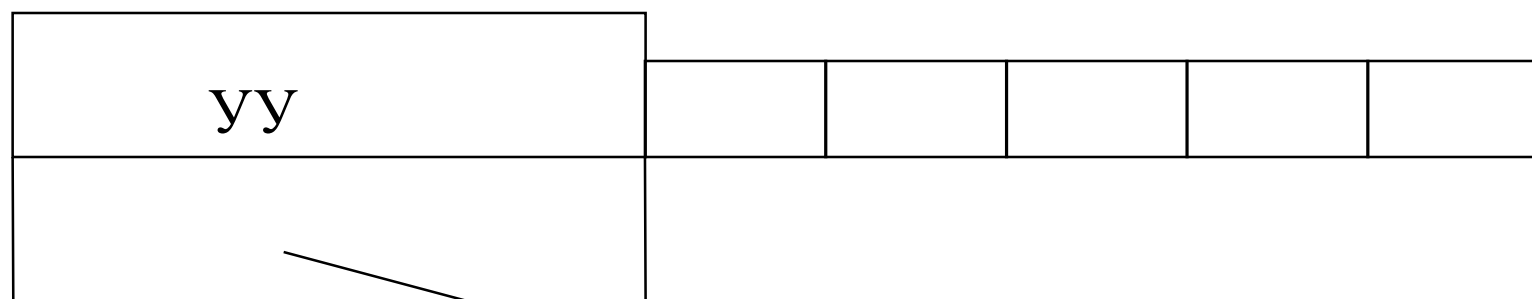
Дефиниция: Нека  $L$  е формален език. **Степените на  $L$**  се определят така:  $L^0 = \{\varepsilon\}$ ;  $L^1 = L$ ;  $L^2 = L.L$ ; ...  
 $L^n = L^{n-1}.L$

Дефиниция: **Итерация  $L^*$**  на произволен език  $L$  ще наричаме обединението на всички степени на  $L$ . Т.е  $L^* = \bigcup L^n, n \geq 0$ .

Итерацията се състои от всички възможни конкатенации на произволен брой думи на  $L$ . Ако всички думи са с дължина 1, тогава  $L^* = V^*$  - всички думи над  $V$ .

# Пораждащи граматики и пораждащи езици

Ще разгледаме един основен вид генератори на формални езици - т. нар. **пораждащи граматики**.  
Да си представим абстрактно устройство от вида:



Състои се от клетки, във всяка от които има по една буква.

# Пораждащи граматики и пораждащи езици

- Управляващото устройство се състои от краен брой вътрешни състояния, като в даден момент се намира само в едно от тях.
- При всяко включване на тока, УУ преминава в едно и също начално състояние.
- След определен брой дискретни тактове на УУ, върху лентата се изписва дума (всеки такт - една буква).
- Машината работи вероятно (т.е. недетерминирано) и генерира думи.

# Пораждащи граматики и пораждащи езици

- За всеки формален език, който може да се опише с крайни средства чрез някакъв автомат, съществува пораждаща го граматика.
- Освен това чрез процеса на пораждане, пораждащите граматики приписват съответна структура на думите от езика, поради което те намират голям брой приложения в математиката и информатиката - от езиците за програмиране и техния автоматичен превод, до моделирането на изчислителните процеси.



# Граматика-генератор (Пораждаща граматика)

Дефиниция: **Граматика-генератор** е наредена четворка от вида:  $\Gamma = \langle V, W, S, P \rangle$ , където:

- $V$  е азбука на терминалните символи (азбука на пораждащия език)
- $W$  е вътрешна нетерминална азбука от вътрешни състояния на УУ (синтактични категории в езика), като  $V \cap W = \emptyset$ . Нетерминалните символи играят роля на синтактични категории в езика
- $S \in W$  е начално състояние

# Граматика-генератор (Пораждаща граматика)

- $P$  е крайно множество от правила на пораждащата граматика, които представляват наредени двойки от думи, съставени от терминални и нетерминални символи  $\langle \alpha, \beta \rangle$  над  $V \cup W$ , като в  $\alpha$  има поне един нетерминален символ (за  $\beta$  не е задължително).

Правилата на пораждащата граматика описват всеки процес на пораждане на изходната терминална дума от началния символ.

# Граматика-генератор

Прието е правилата да се записват във вида  $\alpha \rightarrow \beta$ .

Дефиниция: Думата  $\mu$  **се извежда**

**непосредствено** от думата  $\gamma$  в  $\Gamma = \langle V, W, S, P \rangle$ , ако съществуват думи  $\gamma_1$  и  $\gamma_2 \in (V \cup W)$ , такива че  $\gamma_1 \alpha \gamma_2$ ,  $\gamma_1 \beta \gamma_2$  и съществува правило  $\alpha \rightarrow \beta$ .

Пишем:  $\gamma \vdash \mu$ .

- Когато  $\gamma \vdash \gamma_1 \vdash \gamma_2 \dots \vdash \mu$ , пишем  $\gamma \models \mu$ .

# Граматика-генератор

- Дефиниция: Казваме, че думата  $\alpha$  **се поражда** от граматиката  $\Gamma$ , ако съществува извод  $s \models \alpha$ . Множеството думи, които  $\Gamma$  генерира се наричат **неин език**.

$$L(\Gamma) = \{ \alpha \in V^* : \exists \text{ извод } s \models \alpha \}$$

- Дефиниция: Две граматики са **еквивалентни**, ако  $L(\Gamma_1) = L(\Gamma_2)$ .

# Граматика-генератор

- Пример:  $\Gamma = \langle \{0,1\}, \{s\}, s, \{s \rightarrow 01, s \rightarrow 0s1\} \rangle$

Очевидно  $\Gamma$  е граматика и  $s \vdash 01$ , като  $01 \in L(\Gamma)$ ;

$$\begin{array}{l} \vdash 0^2 1^2 \\ s \vdash 0s1 \vdash 0^2 s 1^2 \vdash 0^3 1^3 \\ \vdash 0^3 s 1^3 \end{array}$$

Т.е.  $L(\Gamma) = \{0^n 1^n; n=1,2..\}$

# Пример

$\Gamma = \langle \{0,1,2,3,4,5,6,7,8,9\}, \{S\}, S, \{S \rightarrow S0, S \rightarrow S1, S \rightarrow S2, S \rightarrow S3, S \rightarrow S4, S \rightarrow S5, S \rightarrow S6, S \rightarrow S7, S \rightarrow S8, S \rightarrow S9, S \rightarrow 1, S \rightarrow 2, S \rightarrow 3, S \rightarrow 4, S \rightarrow 5, S \rightarrow 6, S \rightarrow 7, S \rightarrow 8, S \rightarrow 9\} \rangle$

*Тази граматика поражда десетичните записвания на естествените числа.*

*Например числото 8797 се извежда така:*

$S \vdash S7 \vdash S97 \vdash S797 \vdash 8797$

# Видове граматики и формални езици

**От общ (нулев)тип**- Пораждащи граматики върху правилата, на които не се налагат никакви допълнителни условия, т.е. имат вида  $\alpha A \beta \rightarrow w$  (като  $\alpha, \beta, w$  са думи от терминални и нетерминални символи, а  $A$  е нетерминален символ).

- В този случай  $A$  се замества с  $w$ .

# Видове граматики и формални езици

- **От контекстен тип** - Пораждаща граматика  $\Gamma = \langle V, W, S, P \rangle$ , всички правила на която имат вида:  $\alpha A \beta = \alpha w \beta$ , като  $\alpha, \beta, w \in (V \cup W)^*$   $w \neq \varepsilon$ ,  $A \in W$  се нарича контекстна граматика (или от клас 1), т.е.  $A$  се замества с  $w$  само в даден контекст  $\alpha \rightarrow \beta$ .



# Видове граматики и формални езици

- **От безконтекстен тип**- Пораждаща граматика  $\Gamma = \langle V, W, S, P \rangle$ , всички правила на която имат вида  $A \rightarrow w$  е от тип 2, като  $A$  е от  $W$ ,  $w \in (V \cup W)^*$ .

# Видове граматики и формални езици

- **Автоматна граматика** - Пораждаща граматика  $G = \langle V, W, S, P \rangle$ , чиито правила са от вида :  $A \rightarrow aB$ ,  $A \rightarrow a$ , като  $(A, B \in W, a \in V)$  е автоматна или граматика от тип 3.
- **Дефиниция**: Един формален език е автоматен (тип 3), безконтекстен (тип 2), контекстен (тип 1) или от общ тип (тип 0)  $\Leftrightarrow$  пораждащата го граматика е съответно от тип 3, тип 2, тип 1 или тип 0.

# Примери:

- Граматика с терминални символи  $\{a, b\}$ , нетерминални символи  $\{S, A, B\}$ , правила:

$$S \rightarrow ABS$$

$$S \rightarrow \varepsilon \text{ (където } \varepsilon \text{ обозначава празната дума)}$$

$$BA \rightarrow AB$$

$$BS \rightarrow b$$

$$Bb \rightarrow bb$$

$$Ab \rightarrow ab$$

$$Aa \rightarrow aa$$

и аксиома  $S$ , дефинира генеративно езика на всички думи от вида  $a^n \circ b^n$  (т.е.  $n$  пъти  $a$ , последвани от  $n$  пъти  $b$ ).

- Следната по-проста граматика дефинира генеративно същия език:

Терминални символи  $\{a, b\}$ , нетерминални  $\{S\}$ , аксиома  $S$ , правила:

$$S \rightarrow aSb$$

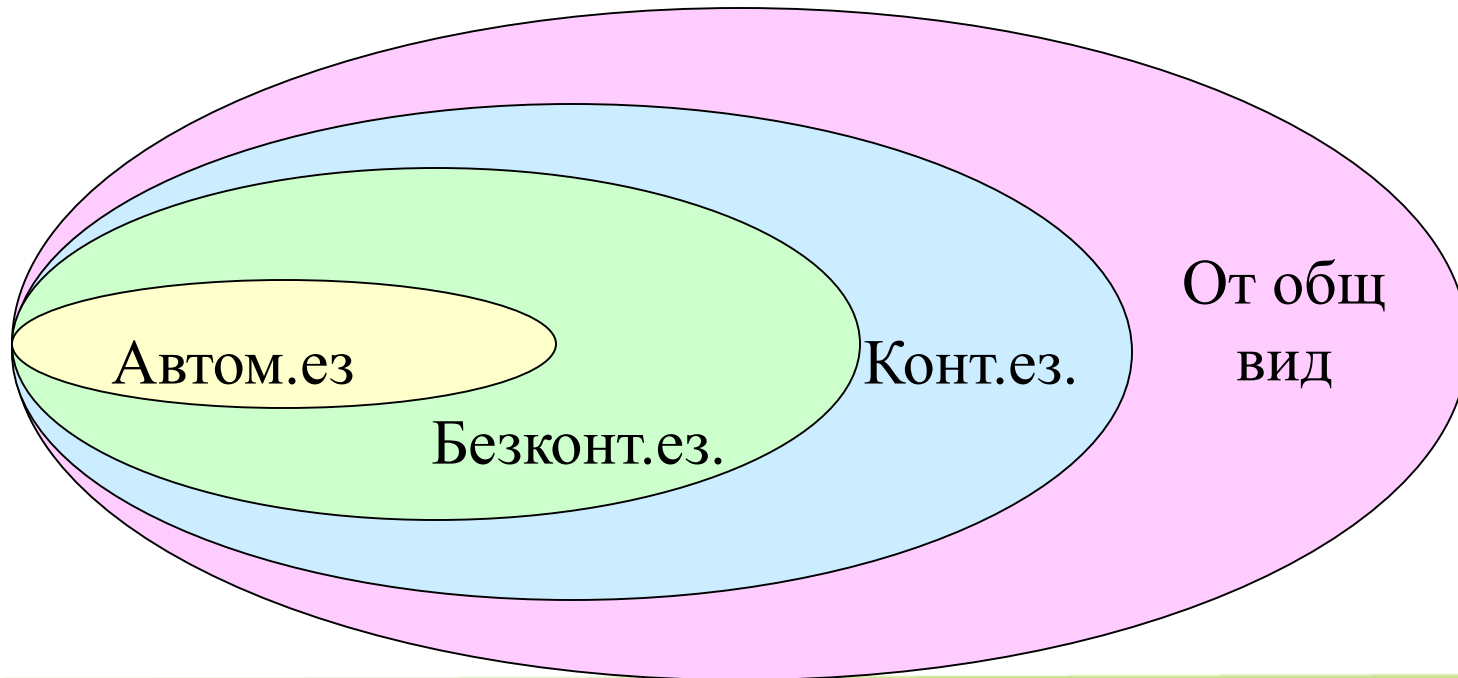
$$S \rightarrow \varepsilon$$

# Йерархията на Чомски

- Това е йерархия от класове формални граматики, образуващи формални езици.
- Въведена е през 1956 г. от американския лингвист Ноам Чомски.
- Освен в лингвистиката, моделът на граматиките на Чомски намира широко приложение и в други науки, като информатиката (тясно свързано със съответствията с концепти от теорията на автоматите) и биологията (Нилс К. Йерне озаглавява нобеловата си лекция „Генеративната граматика на имунната система“ и разглежда протеиновия строеж в такъв контекст).

# Йерархия на формалните езици

- Автоматните, безконтекстните, контекстните и езиците от общ вид образуват йерархията на Чомски за формалните езици



# Регулярни езици

- Множеството на регулярните езици е равно на множеството на езиците, разпознавани от крайни автомати, т.е. всеки език, разпознаван от краен автомат, е регулярен (теорема на Клини (Kleene)). Това означава, че всеки регулярен израз може да се представи като краен автомат и обратното.

# Абстрактни машини

Типовете в йерархията съответстват на езиците, разпознавани от различни видове абстрактни машини:

Грама- тика	Език	Автомат
Тип 0	рекурсивно изброим	Машина на Тюринг
Тип 1	контекстен	линейно ограничена недетерминирана машина на Тюринг
Тип 2	Безконтекстен	Магазинен автомат
Тип 3	регулярен	Краен автомат

# Твърдения

- **Лема:** Нека  $L$  е формален език от тип  $I$  ( $I=0..3$ ).  
Тогава  $L \cup \{\epsilon\}$  и  $L - \{\epsilon\}$  са формални езици от същия тип.
- **Теорема:** Нека  $L_1$  и  $L_2$  са произволни автоматни езици. Тогава  $L_1 \cup L_2$  е също автоматен език.
- **Теорема:** Нека  $L_1$  и  $L_2$  са произволни автоматни езици. Тогава  $L_1.L_2$  е също автоматен език.
- **Теорема:** Всеки краен език е автоматен. (например-българския език.)



# Използвана литература в курса

- D. W. Hoffmann, Theoretische Informatik, Hansen Verlag, 2009
- H. P. Gumm, M. Sommer, Einfuehrung in die Informatik, Oldenbourg Wissenschaftsverlag, 2004
- J. W. Grossman, Discrete Mathematics, Macmillan Pub. Co., 1990
- К. Манев, Увод в дискретната математика, КЛМН, 2005
- Й. Денев, Р. Павлов, Я. Демирович. *Дискретна математика*. Наука и изкуство, София, 1984.

# Използвана литература в курса

- Д. Байнов, С. Костадинов, Р. Павлов, Л. Луканова. *Ръководство за решаване на задачи по дискретна математика*. Университетско издателство "Паисий Хилендарски", Пловдив, 1990.
- В.А. Успенский, *Машина Поста*, Москва, Наука, 1988, ISBN 5-02-013735-9.
- L. Lovasz, J. Pelikan, K. Vesztergombi, *Discrete Mathematics – Elementary and Beyond*, Springer Verlag, New York, 2003, ISBN 0-387-95584-4.

# Исползвана литература в курса

- E. Bender, S. Williamson, *A Short Course in Discrete Mathematics*, Dover, 2006, ISBN 0-486-43946-1.
- P. Linz, *An Introduction to Formal Languages and Automata*, Jones and Bartlett Publishers, 6-th edition, Jones & Bartlett Publishers, ISBN-13: [9781284077247](#), 2016
- Kenneth H. Rosen, Kamala Krithivasan, *Discrete mathematics and its application*, McGraw-Hill Companies, 7-th edition, ISBN 978-0-07-338309-5, 2012

# Използвана литература в курса

- Owen D. Byer, Deirdre L. Smeltzer, Kenneth L. Wantz, Journey into Discrete Mathematics, AMS, MAA Press, Providence Rhode Island, ISBN 9781470446963, 2018
- Christopher Rhoades, Introductory Discrete Mathematics, Willford Press, ISBN 1682854922, 9781682854921, 2018
- David Liben-Nowell, Discrete Mathematics for Computer Science, Wiley, 2017, ISBN 1119397197, 9781119397199, 2017.
- <http://www.jflap.org/> - софтуерна среда