1. Графи. Пътища.

Определение 1 - *Неориентиран граф (граф)* наричаме наредена двойка $G = \langle V, E\rangle$ от множества, където $V = \{v_1, \ldots, v_n\} \neq \emptyset$ е множество от *върхове*, а $E \subset V^2$ е множество от ненаредени двойки върхове - $e = (v_i, v_j) \in E$ наречени *ребра*. Ще разглеждаме крайни графи, т.е. $|V| < \infty$ и $|E| < \infty$.
Пример 1) $V = \{v_1, v_2, v_3\}$, $E = \{(a_1, a_2), (a_1, a_3), (a_2, a_3)\}$ - чертеж;
2) празен граф; 3) паралелни ребра; 4) примка;
Определение 2 - Два върха наричаме *съседни*, ако са краища на ребро. Броят на ребрата с край върха v_i наричаме *степен* (бележим $\deg(v_i)$).

Връх свързан със себе си наричаме *самосвързан*. Две ребра са *паралелни*, ако свързват едни и същи върхове.
Определение 3 - Граф, в който няма примки и паралелни ребра наричаме *прост граф*.

За простите графи е в сила равенството

$$|E| = \frac{1}{2}\sum_i \deg(v_i).$$

Определение 4 - *Ориентиран граф* наричаме граф, в който ребра са наредени двойки $= \langle v_i, v_j\rangle$, v_i– начало на реброто, v_j – край.
Пример 2) $V = \{v_1, v_2, v_3\}$, $E = \{(a_1, a_2), (a_1, a_3), (a_2, a_3)\}$ - чертеж;
Определение 5 - Път в графа $G = \langle V, E\rangle$ съединяващ v_1 с v_{n+1} ще наричаме редица от вида $v_1e_1v_2e_2 \ldots v_ne_nv_{n+1}$, където $e_i = (v_i, v_{i+1}) \in E$. Броят на ребрата в пътя наричаме *дължина* на пътя. Път, при който всички ребра са различни наричаме *прост* път.
Определение 6 - Ако един път няма нито повтарящи се върхове, нито повтарящи се ребра, то този път се нарича *верига*. Всяка затворена верига наричаме *цикъл*.

Теорема 1. *Ако между два върха в един граф съществува път, който ги свързва, то между тях съществува и прост път, който ги свързва.*

Доказателство. Ако $v_1e_1 \ldots e_{n-1}v_n$ е произволен път, то:
1) ако пътя е прост, то няма какво да доказваме;
2) ако пътят не е прост, премахваме всички цикли и получаваме прост път. □

Теорема 2. *Ако $G = \langle V, E\rangle$ е краен граф с n върха, то следните твърдения са еквивалентни.*
а) съществува път в G с дължина по-голяма или равна на n;
б) в G съществува цикъл.

Доказателство. 1)⇒2) Ако $a_0e_1a_1e_2 \ldots e_sa_s$ е път с дължина $s \geq n$, съществуват два еднакви върха в пътя, т.е. $k \neq l$ и $a_k = a_l$. Тогава пътят $a_ke_{k+1} \ldots e_la_l$ е цикъл в графа.
2)⇐1) е очевидно □

Определение 7 - Един граф наричаме *свързан*, ако през всеки негови два върха има път.

Нека $G = \langle V, E\rangle$ е граф и $x \in E$, под $G \setminus \{x\} = \langle V, E \setminus \{x\}\rangle$.

Теорема 3. *Ако графът G е свързан и реброто x е в цикъл, то $G \setminus \{x\}$ също е свързан.*

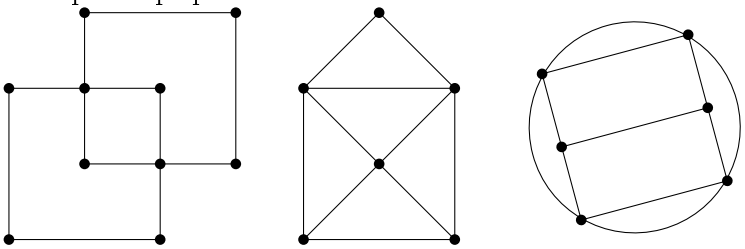
Определение 8 - Всеки прост път, който съдържа всички ребра на графа се нарича *Ойлеров път* в G. Намирането на Ойлеров път в един граф е една от първите задачи в теорията на графите.

Теорема 4. *Свързаният граф G има затворен Ойлеров път тогава и само тогава, когато всичките му върхове имат четна степен.*

Следствие 1. *Свързаният граф G има отворен Ойлеров път тогава и само тогава, когато G има точно два върха от нечетна степен. За да го обходим започваме от единия нечетен връх и трябва да завършим в другия.*

Определение 9 - Граф, който има Ойлеров път се нарича Ойлеров граф.

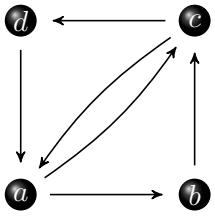
Ойлерови графи



2. Матрица на съседство. Оптимални пътища

Нека G е ориентиран граф с n върха. Да номерираме върхове на графа с числата $1, 2, \ldots, n$. Матрицата $A = (c_{ij})$ с размери $n \times n$ се нарича *матрица на съседство*, ако c_{ij} е равно на броя на ребрата, които водят от върха a_i до върха a_j.

Пример: Следният граф има матрица на съседство $A = \begin{pmatrix} 0\,1\,1\,0 \\ 0\,0\,1\,0 \\ 1\,0\,0\,1 \\ 1\,0\,0\,0 \end{pmatrix}$



Да повдигнем матрицата на съседство $A = (c_{ij})$ на квадрат. Получаваме $A^2 = (c_{ij}^2)$, където

$$c_{ij}^2 = \sum_{k=1}^{n} c_{ik} c_{kj}.$$

Числото c_{ik} дава броя на пътищата с дължина 1 (т. е. на ребрата), които водят от a_j до a_k, а $c + kj$ - броя на пътищата с дължина 1, които водят от a_k до $_j$. Очевидно $c_{ik}c_{kj}$ е броят на пътищата с дължина 2, които водят от a_i до a_j и минават през a_k. Тъй като сме извършили сумиране по всички $k = 1, \ldots, n$, то c_{ij}^2 е броят на всички пътища, a_i до a_j и имат дължина 2.

С индукция може да се докаже и следната теорема.

Теорема 5. *Нека $A = (c_{ij})$ е матрица на съседство на графа G. Тогава елементът c_{ij}^n е равен на броя на пътищата с дължина n, които водят от a_i до a_j.*

Необходимо и и достатъчно условие в G да има цикъл е $A^n \neq 0$. Наистина, ако $A^n \neq 0$, то в графа има път с дължина n и съгласно теоремата - цикъл. Обратно, ако в графа има цикъл, то в него има път с дължина n и следователно $A^n \neq 0$.

Да означим с $r(a_i, a_j)$ дължината на на най-късия път, който води от върха a_i до върха a_j. Свойствата на матрицата за съседство A ни дават един прост начин за определяне $r(a_i, a_j)$. За тази цел повдигаме на степен, докато за първи път елементът c_{ij}^l на A^l стане различен от нула. Тогава $r(a_i, a_j)$.

Пресмятането на степен на матрица обаче не е лесно. Един от най-простите алгоритми за определяне на $r(a_i, a_j)$ може да се опише по следния начин:

1. Полагаме $k = 0$ и $S_0 = \{i\}$.
2. $S_k = \left\{ \begin{array}{l} \text{всички върхове } a_p \text{ за които} \\ r(a_p, b_q) = 1, b \in S_{k-1} \end{array} \right\} \setminus \{S_0 \cup S_1 \cup \cdots \cup S_{k-1}\}$
3. Ако $a_j \in S_k$, разстоянието $r(a_i, a_j) = k$, а ако $a_j \notin S_k$, то полагаме $k = k + 1$ и преминаваме към 2.

3. Дървета

Определение 1 - Свързан граф без цикли ще наричаме *дърво.*

От определението получаваме, че всеки два върха в дърво могат да се свържат с път и при това този път е единствен. Следната теорема може да се използва като определение на понятието дърво.

Теорема 6. *Един граф е дърво тогава и само тогава, когато между всеки два върха има точно един път.*

Определение 2 - Ако един връх има степен 1, то този връх се нарича *краен.*

Теорема 7. *Всяко дърво, което има поне едно ребро има поне два крайни върха.*

Теорема 8 (Доказателство)**.** *Дървото има краен брой върхове, следователно съществува верига с максимална дължина. В тази верига първия и последният връх имат степен 1, т.е. са крайни.*

Теорема 9. *Всяко дърво с n върха има $n-1$ ребра.*

Доказателство. Ще използваме индукция.

1. $n = 1$ - имаме връх без ребра, т.е. с $1-1$ ребра - вярно;
2. $n > 1$. Допускаме, че твърдението е вярно за всяко дърво с по-малко от n върха;
3. Разглеждаме дърво с точно n върха. Нека v е краен връх. Премахваме върха v и единственото ребро, което го свързва. Получаваме нов граф, в който няма цикли, т.е. имаме дърво с $n-1$ върха, което според 2) има $n-2$ ребра. Тогава изходното дърво е имало $n-1$ ребра. □

Определение 3 - Нека G е свързан граф. Всяко дърво, което е подграф на G и съдържа всичките му върхове ще наричаме *покриващо дърво* на графа G.

Теорема 10. *Всеки свързан граф има поне едно покриващо дърво.*

Доказателство. Индукция по броя ребра в графа G. Ако има едно ребро, то си е прокриващо дърво. Допускаме, че теоремата е доказана за всички графи с $n-1$ ребра. Нека G е свързан граф с n ребра. Ако в него няма цикли, то той е дърво и следователно си е покриващо дърво. Нека графът не е дърво, т.е. има цикъл. Избираме ребро v от този цикъл и разглеждаме $G' = G \setminus \{v\}$ - граф, който има $n-1$ ребра, а следователно и покриващо дърво, но G' има точно върховете, които и G. Тогава покриващото дърво за G' е покриващото дърво и за G . □

Следствие 2. *Всеки граф с n върха и $n-1$ ребра е дърво.*

Доказателство. Допускаме противното и разглеждаме покриващо дърво D. Тогава D има n върха и $\leq n-2$ ребра и понеже е дърво според по-горна теорема има $n-1$ ребра - противоречие. □

4. Двоични функции начини на задаване, свойства

Да означим $\mathcal{B}$ множеството $\{0,1\}$. Да положим $\mathcal{B}^n = \mathcal{B} \times \cdots \times \mathcal{B}$, където дясната страна на равенството е декартовото произведение на n множества $\mathcal{B}$. Очевидно $\mathcal{B}$ се състои от всички подредени n-орки, съставени от нули и единици. Техният брой е 2^n. Елементите на $\mathcal{B}^n$ ще наричаме двоични n-орки.

Определение 1- Под двоична функция $f : \mathcal{B}^n \to \mathcal{B}$ във на n променливи ще разбираме функция с област $\mathcal{B}^n$ и кообласт $\mathcal{B}$.

Теорема 11. *Броят на всички двоични функции на n променливи е* 2^{2^n}.

Доказателство. Имаме $|\mathcal{B}| = 2^n$ различни n-орки, като за всяка от тях можем да даваме по 2 различни стойности - 0 и 1. Тогава броят на различните двоични функции на n променливи е наистина 2^{2^n}. □

Определение 2 - Ще казваме, че променливата x_i е *фиктивна или несъществена* за функцията f, ако е изпълнено

$$f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n).$$

Ако една променлива не е фиктивна, то ще я наричаме *съществена.*

По-нататък ще разглеждаме само функции, в които всички променливи са съществени. Според вече доказаната теорема имаме 4 двоични функции на една променлива и 16 на 2.

x_1	0	x_1	$\overline{x_1}$	1
0	0	0	1	1
1	0	1	0	1

Първата и четвъртата функция са функции-константи – за тях променливата x_1 е фиктивна. Тези функции означаваме съответно с 0 и 1 и съгласно нашата уговорка не ги различаваме от двоичните константи 0 и 1, които може да се разглеждат като функции на нула променливи. Втората функция означаваме с x_1 и наричаме тъждествена функция. Третата функция означаваме с $\overline{x_1}$ и наричаме отрицание. Веднага се проверява, че е в сила свойството $\overline{\overline{x_1}} = x_1$, което наричаме закон на двойното отрицание.

Шестнадесетте функции на две променливи са зададени в следната таблица.

x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

От тях ще отбележим следните функции:

f_0, която е тъждествено равна на 0, при нея променливите x, и y са фиктивни и ние няма да различаваме тази функция от нулата, разглеждана като функция на 0 или 1 променлива;

f_{15}, която е тъждествено равна на 1 и може да се разглежда аналогично на 1;

f_1, която ще означаваме с xy и ще наричаме *конюнкция*; за същата функция се използват понякога означенията $x \wedge y$ и $x\&y$, както и наименованията *логическо умножение и логическо "и"*;

f_3, която съвпада с тъждествената функция x;

f_5, която съвпада с тъждествената функция y;

f_6, която ще означаваме с $x + y$, и ще наричаме *сума по модул 2*; същата функция се означава понякога с $x \oplus y$ и се нарича *изключващо "или"*;

f_7, която ще означаваме с $x \vee y$ н ще наричаме *дизюнкция (или логическо "или")*;

f_8, която ще означаваме с $x_1 \downarrow x_2$ н ще наричаме *функция (или стелка) на Пирс*;

f_{10}, която ще означаваме с $x \equiv y$, и ще наричаме *еквивалентност*;

f_{13}, която ще означаваме с $x \to y$, и ще наричаме *импликация*;

f_{15} която ще означаваме с $x|y$ и ще наричаме *функция (или черта) на Шефер.*

Ето някои от свойствата на тези функции:

1. $xx = x$, $x \vee x = x$, $x + x = 0$;

2. $xy = yx$, $x \vee y = y \vee x$, $x + y = y + x$;

3. $x(yz) = (xy)z$, $x \vee (y \vee z) = (x \vee y) \vee z$, $(x + y) + z = x + (y + z)$;

4. $x(y \vee z) = (xy) \vee (xz)$, $x \vee (yz) = (x \vee y)(x \vee z)$, $x(y + z) = (xy) + (xz)$;

5. $x0 = 0x = 0$, $x \vee 0 = x$, $x + 0 = x$;

6. $x1 = x$, $x \vee 1 = 1$, $x + 1 = \overline{x}$;

7. $x\overline{x} = 0$, $x \vee \overline{x} = 1$, $x + \overline{x} = 1$;

8. $\overline{(xy)} = (\overline{x}) \vee (\overline{y})$, $\overline{(x \vee y)} = (\overline{x})(\overline{y})$.

Първите две равенства в т. 1 по-горе се наричат закони за идемпотентност, свойствата от т. 2 – закони за комутативност, свойствата от т. 3 - за асоциативност, от т. 4 – за дистрибутивност и накрая от т. 8 – закони на Де Морган.

Пример: Да установим верността на втория закон на Де Морган. За целта съставяме таблицата

x	y	$x \vee y$	$\overline{x}$	$\overline{y}$	$\overline{x \vee y}$	$\overline{x}\,\overline{y}$
0	0	0	1	1	1	1
0	1	1	1	0	0	0
1	0	1	0	1	0	0
1	1	1	0	0	0	0

Приоритет на изпълнение:

а) подизрази, заградени в скоби;

б) отрицания;

в) конюнкция;

г) дизюнкция и сума по модул 2;

д) всички останали функции.

задачи: Докажете тъждествата: а) 1,3,5.

б) 2,4,6.

5. Пълни множества от двоични функции. Полиноми на Жегалкин

Определение 1 - Ще разваме, че множеството от двоични функции F е пълно, ако всяка двоична функция се реализира с формула над F, т.е. $[F] = P_2$ - множеството на всички двоични функции.

Теорема 12 (Бул). *Множеството от двоични функции $\{\overline{x}, x \vee y, xy\}$ (отрицание, конюнкция и дизюнкция) е пълно.*

Доказателство. Нека функцията, е константната 0, тогава $0 = x\overline{x}$. Нека сега $f(x_1, \ldots, x_n) \neq 0 \in P_2$. Да означим $x^a = \begin{cases} \overline{x}, & \text{a=0;} \\ x, & \text{a=1.} \end{cases}$ Не е трудно да се види, че $x^a = 1$ тогава и само тогава, когато $x = a$. Оттук $x_1^{a_1} \ldots x_n^{a_n} = 1$ тогава и само тогава, когато $x_i = a_i$, $i = 1, \ldots, n$ и

$$f(x_1, \ldots, x_n) = \bigvee_{f(a_1,\ldots,a_n)=1} x_1^{a_1} \ldots x_n^{a_n},$$

където дизюнкцията се взима по всички n-торки, за които функцията приема стойност 1.

Наистина, нека $f(b_1, \ldots, b_n) = 1$. Тогава в дясната страна има член на дизюнкцията $b_1^{b_1} \ldots b_n^{b_n} = 1$, откъдето получаваме, че самата дизюнкция ще е равна на 1. Ако $f(b_1, \ldots, b_n) = 0$, във всички членове на дизюнкцията ще има поне един множител, чиято стойност няма да съвпадне със степента си, а оттам всички членове, а следователно и самата дизюнкция ще е равна на 0. □

Теорема 13. *Нека $F = \{f_1(x_1, \ldots, x_n), f_2(x_1, \ldots, x_n), \ldots\}$ е пълно множество от двоични функции. Тогава необходимо и достатъчно условие $G = \{g_1(x_1, \ldots, x_n), g_2(x_1, \ldots, x_n), \ldots\}$ да е пълно е всички функции $f_1, \ldots, f_l$ от F да се изразяват чрез $g_1, g_2, \ldots$*

Доказателство. Необходимостта е очевидна.

За достатъчността, нека $\varphi_1, \ldots, \varphi_l, \ldots$ са формули над G, които реализират $f_1, \ldots, f_l$. Ако изберем произволна функция $h(x_1, \ldots, x_n)$ и да я реализираме с формула ψ над F. Да заменим в ψ елементите $f_1, \ldots$ с $\varphi_1, \ldots$. Тогава ще получим формула над G, която реализира функцията $h(x_1, \ldots, x_n)$. □

От тези две теореми следва пълнотата на редица множества:

1. $\{x \vee y, \overline{x}\}$, т.к. $xy = \overline{\overline{x} \vee \overline{y}}$;
2. $\{xy, \overline{x}\}$, т.к. $x \vee y = \overline{\overline{x}\,\overline{y}}$;
3. $\{x|y\}$, т.к. $\overline{x} = x|x$, $x \vee y = (x|x)|(y|y)$;
4. $\{x \downarrow y\}$, т.к. $\overline{x} = x \downarrow x$, $xy = (x \downarrow x) \downarrow (y \downarrow y)$;
5. $\{xy, x + y, 1\}$, т.к. $\overline{x} = x + 1$.

Да разгледаме по-подробно пълното множество $\{xy, x + y, 1\}$. Не е трудно да се види, че всяка формула над това множество може да се запише във вида

$$M_1 + M_2 + \cdots + M_i,$$

където M_i е едночлен от вида $x_{j_1} \ldots x_{j_k}$ или константа. Използвайки $xx = x$ всеки едночлен M_i съдържа само различни неизвестни, а чрез $M_i + M_i = 0$ тази сума може да се сведе до сума от различни събираеми.

Определение 2 - Израз от вида $E_1 + \cdots + E_k$, в който едночлените са различни и всеки от тях $E_i = x_{j_1} \ldots x_{j_k}$ е произведение от различни неизвестни или е константа се нарича *полином на Жегалкин.*

Да преброим полиномите на Жегалкин на n променливи. Броят на едночлените с j различни променливи е $\binom{n}{j}$, $j = 1, \ldots, n$ и следователно броят на всички различни едночлени е

$$\sum_{i=1}^{n} \binom{n}{i} = \sum_{i=1}^{n} \binom{n}{i} 1^i 1^{n-i} = (1+1)^n = 2^n.$$

Всеки различен полином на Жегалкин, очевидно задава различна двоична функция, но броят на различните полиноми е точно 2^{2^n}, точно колкото са всички различни двоични функции на n променливи. Тогава доказахме следната теорема.

Теорема 14. *Всяка двоична функция се реализира точно с един полином на Жегалкин.*

6. Затворени класове T_0, T_1, S. Монотонни и линейни функции. Критерий за пълнота.

Определение 1 - Едно множество от двоични функции C наричаме *затворено или затворен клас*, ако $C = [C]$.

Теорема 15. *Множеството C е затворен клас, ако:*
а) тъждествената функция x е от C;
б) една сложна функция $f(g_1, \ldots, g_n) \in C$, ако $f, g_1, \ldots, g_n \in C$.

Определение 2 - Ще казваме, че $f(x_1, \ldots, x_n)$ *запазва нулата*, ако $f(0, \ldots, 0) = 0$. Множеството от всички двоични функции, които запазват нулата ще означим с T_0.
Определение 3 - Ще казваме, че $f(x_1, \ldots, x_n)$ *запазва единицата*, ако $f(1, \ldots, 1) = 1$. Множеството от всички двоични функции, които запазват 1 ще означим с T_1.

Тъй като една функция от T_0 или T_1 фиксира стойност за една наредена n-торка, то очевидно $|T_0| = |T_1| = 2^{2^{n-1}}$.

Теорема 16. *Множествата T_0 и T_1 са затворени.*

Доказателство. Ще докажем за T_0 (доказателството за T_1) е аналогично. Имаме, че $x_i \in T_0$ и ако $f, g_1, \ldots, g_n \in T_0$, то $f(g_1(0, \ldots, 0), g_2(0, \ldots, 0), \ldots, g_n(0, \ldots, 0)) = f(0, \ldots, 0) = 0$ и сложната функция f също се оказа от T_0. □

Теорема 17. *Множествата T_0 и T_1 са затворени.*

Доказателство. Ще докажем за T_0 (доказателството за T_1) е аналогично. Имаме, че $x_i \in T_0$ и ако $f, g_1, \ldots, g_n \in T_0$, то $f(g_1(0, \ldots, 0), g_2(0, \ldots, 0), \ldots, g_n(0, \ldots, 0)) = f(0, \ldots, 0) = 0$ и сложната функция f също се оказа от T_0. □

Теорема 18. *Нека $f_0 \notin F_0$, $f_1 \notin F_1$. Тогава с формули над $\{f_0, f_1\}$ можем да построим константите 0 и 1 или отрицанието, т.е. $\{0, 1\} \subseteq \{f_0, f_1\}$ или $\{\overline{x}\} \subseteq \{f_0, f_1\}$.*

Доказателство. Да положим $g_0 = f_0(x_1, \ldots, x_1)$, $g_1 = f_1(x_1, \ldots, x_1)$, Очевидно $g_0(0) = g_0(0, \ldots, 0) = 1$, $g_1(1) = 0$. Нека $g_0(1) = a$, $g_1(0) = b$. Имаме четири случая за a и b:
1. $a = 0, b = 0$. Тогава $g_0 = \overline{x_1}, g_1 = 0$.
2. $a = 0, b = 1$. Тогава $g_0 = \overline{x_1}, g_1 = \overline{x_1}$.
3. $a = 1, b = 0$. Тогава $g_0 = 1, g_1 = 0$.
4. $a = l, b = 1$. Тогава $g_0 = 1, \overline{x_1}$.

В случая 2 получихме отрицанието, в случая 3 - константите, а в случаите 1 и 4 – отрицанието и константите. □

Определение 4 - Функцията $\overline{f(\overline{x_1}, \ldots, \overline{x_n})}$ ще наричаме *двойнствена* на f и ще я означаваме с $f^*(x_1, \ldots, x_n)$.

Пример: 1. $(\overline{x})^* = \overline{\overline{x}} = \overline{x}$; 2. $x^* = \overline{\overline{x}} = x$; 3. $(xy)^* = \overline{\overline{x}\,\overline{y}} = x \vee y$.

Теорема 19. *Двойнствената функция $(f(g_1, \ldots, g_n))^*$ на сложна функция е сложна функция на двойнствените и компоненти, т. е.*

$$(f(g_1, \ldots, g_n))^* = f^*(g_1^*, \ldots, g_n^*).$$

Теорема 20 ((принцип на двойнственоста)). *Нека F е множество от двоични функции $F = \{f_1(x_1, \ldots, x_n), f_2(x_1, \ldots, x_n), \ldots, \}$, а G - множеството на двойнствените функции на функциите от F:*

$$G = \{f_1^*(x_1, \ldots, x_n), f_2^*(x_1, \ldots, x_n), \ldots, \}$$

Тогава, ако в произволна формула Φ над F заменим символите на функциите $f_1, f_2, \ldots$ с $f_1^*, f_2^*, \ldots$ ще получим формула ψ над G, която ще реализира двойнствената функция на функцията реализирана от Φ.

Определение 5 - Ще казваме, че $f(x_1, \ldots, x_n)$ е самодвойнствена, ако $f^*(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$. Множеството на самодвойнствените функции ще означим с S.

Теорема 21. *Множеството S е затворен клас.*

Теорема 22. *Константите 0 и 1 са суперпозиции над множеството f_0, f_1, f_s където $f_0 \notin F_0$, $f_1 \notin F_1$, $f_s \notin S$ $\{0,1\} \subseteq [f_0, f_1, f_s]$.*

Доказателство. В теорема показахме, че с f_0 и f_1 можем да построим или константите 0 и 1, или отрицанието $\overline{x}$. Ако сме построили константите, исканата конструкция е получена. Да предположим, че с f_0 и f_1 сме построили само отрицанието. Тъй като f_s не е само двойнствена, то съществува n-орка за която

$$f_s(a_1, \ldots, a_n) = f_s(\overline{a_1}, \ldots, \overline{a_n}).$$

Да положим $g(x_1) = f_s(x_1^{a_1} x_1^{a_2} \ldots x_1^{a_n})$. Тази функция се получава чрез функциите от $[f_0, f_1, f_s]$ и имаме

$$g(0) = f_s(0^{a_1}, \ldots, 0^{a_n}) = f_s(\overline{a_1} \ldots \overline{a_n}) = f_s(a_1, \ldots, a_n) = f_s(1^{a_1}, \ldots, 1^{a_n}) = g(1),$$

т.е. g е константа. Използвайки отрицанието $\overline{g}$ можем да получим и другата константа. □

Определение 6 - Нека $a = (a_1, \ldots, a_n)$ и $a = (b_1, \ldots, b_n)$ са две n-орки. Ще казваме, че a *предхожда* b (или че b *следва* a), ако са изпълнени неравенствата

$$a_1 \leq b_1, a_2 \leq b_2, \ldots, a_n \leq b_n.$$

Факта, че a предхожда b ще означаваме с $a \prec b$.

Пример: $(0,0,0,0) \prec (0,0,1,0) \prec (1,0,1,0) \prec (1,1,1,1)$.

Определение 7 - Ще казваме, че $f(x_1, x_2, \ldots, x_n)$ е *монотонна*, ако от $a \prec b$ следва $f(a) \prec f(b)$.

Пример: Лесно се установява с непосредствена проверка, че xy и $x \vee y$ са монотонни. Функциите $x + y$ и $\overline{x}$ не са монотонни, защото $0 + 1 > 1 + 1$ и $\overline{0} > \overline{1}$.

Теорема 23. *Множеството M е затворен клас.*

Теорема 24. *Нека $f_0 \notin F_0$, $f_1 \notin F_1$, $f_s \notin S$, $f_m \notin M$. Тогава функциите $0, 1, \overline{x}$ са суперпозиции над $\{f_0, f_1, f_s, f_m\}$.*

Ще казваме, че $f(x_1, \ldots, x_n)$ е линейна, ако нейният полином на Жегалкин е линеен, т.е. от вида $a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_n x_n$. Да означим с L множеството от всички линейни функции.

Теорема 25. *Множеството L е затворен клас.*

Теорема 26. *Нека $f_0 \notin F_0$, $f_1 \notin F_1$, $f_s \notin S$, $f_m \notin M$, $f_l \neq inL$. Тогава функциите $0, 1, \overline{x}, xy$ са суперпозиции над $\{f_0, f_1, f_s, f_m, f_l\}$.*

Теорема 27 (Критерий за пълнота)**.** *Необходимото и достатъчно условие множеството от двоични функции F да е пълно е да не се съдържа изцяло в нито едно от множествата T_0, T_1, S, M и L.*

Доказателство. Необходимост. Да предположим, че F е пълно множество и се съдържа изцяло например в T_0. Тогава за множеството от всички двоични функции имаме $P_2 = [F] \subset [T_0] = T_0 \neq P_2$, което е невъзможно.

Достатъчност. Тъй като в F има функции $f_0 \notin F_0$, $f_1 \notin F_1$, $f_s \notin S$, $f_m \notin M$, $f_l \neq inL$, то от предходната теорема следва, че пълната система $\{xy, \overline{x}\}$ се реализира с формули над F. □

Задачи: Чрез критерия за пълнота докажете, че следните множества двоични функции са пълни:

1. $\{x|y\}$.
2. $\{x \downarrow y\}$.

7. Минимизация на двоични функции. Дизюнктивна нормална форма

Ще разглеждаме само формули над пълното множество $\{xy, x \vee y, \overline{x}\}$. Всяка функция от P_2 се реализира с безбройно много формули над това множество. Например ако Φ е формула за дадена функция f, то $\Phi \vee x_1\overline{x_1}$ и $\Phi \vee x_1\overline{x_1} \vee x_2\overline{x_2}$ са различни формули, но реализират една и съща функция f.

Определение 1 - Под *сложност на формула* ще разбираме броя на буквите на променливите в нея.

От безбройно многото формули, които реализират дадена двоична функция f ние се интересуваме само от тези с минимална сложност.

Определение 2 - Израз от вида $x_{i_1}^{a_1} \dots x_{i_k}^{a_k}$ където буквите на променливите са различни, както и константата 1 ще наричаме *елементарна конюнкция*.

Определение 3 - Дизюнкция на различни елементарни конюнкции ще наричаме *дизюнктивна нормална форма (ДНФ). Минималната ДНФ* (МДНФ) на функцията f е дизюнктивна нормална форма на f с минимална сложност.

Нека $f(x_1, \dots, x_n)$ е двоична функция на n променливи. Тогава за всяка наредена n-торка, $(a_1, \dots, a_n)$, за която $f(a_1, \dots, a_n) = 1$ да разгледаме

$$x_1^{a_1} x_2^{a_2} \dots x_n^{a_n} = \begin{cases} 0, & (x_1, \dots, x_n) \neq (a_1, \dots, a_n); \\ 1, & (x_1, \dots, x_n) = (a_1, \dots, a_n); \end{cases} .$$

Тогава очевидно

$$f(x_1, \dots, x_n) = \bigvee_{(a_1,\dots,a_n): f(a_1,\dots,a_n)=1} x_1^{a_1} x_2^{a_2} \dots x_n^{a_n}.$$

Тази формула наричаме *съкратена ДНФ (СкДНФ)* .

Определение 4 - Съкратената ДНФ, която има минимална сложност наричаме *съвършенна ДНФ (СвДНФ)*.

Пример: Да разгледаме функцията $x \equiv y$, за която

x	y	$x \equiv y$
0	0	1
0	1	0
1	0	0
1	1	1

и СДНФ е $x^0y^0 \vee x^1y^1 = \overline{x}\,\overline{y} \vee xy$

Алгоритъм на Куайн-Макласки за минимизиране на ДНФ. За него използваме следната формула

$$x_iK \vee \overline{x_i}K = K,$$

наречена принцип на слепването.

Нека $f(x_1, \dots, x_n)$ е произволна функция на n променливи. Методът на Куайн-Макласки се състои в последователно слепваве на конюнкции, като се започва от конюикциите на съвършената ДНФ. Този процес се отразява в таблица

Най–левият стълб (n) се състои от всички импликанти на f с n букви на променливи (т. е. от импликантите на СвДНФ). Извършваме всички възможно слепвания в най-левия стълб и резултатите от слепванията записваме в следващия стълб $(n-1)$. Импликантите от стълба $(n-1)$ се състоят от по $n-1$ букви на променливи. Всички ипликанти от стълба (n), конто са участвали в поне едно слепване маркираме със *. По същия начин слепваме и маркираме импликантите от стълба $(n-1)$. Получаваме стълба $(n-2)$ с всички импдиканти на 1 с $n-2$ букви. Този процес на слепване и маркиране продължава, докато получим стълба (r), в който не можем да направим никакво слепване.

Теорема 28. *В стълбовете* (n), $(n-l)$, $\dots$, (r) *се съдържат всички ипликанти на* f. *Всички неотбелязани със * импликанти са прости и участват в МДНФ.*

Пример: **Пример:**Да намерим СвДНФ на функцията $x_1x_2x_3x_4 \vee x_1x_2x_3\overline{x_4} \vee x_1x_2\overline{x_3}x_4 \vee x_1\overline{x_2}x_3x_4 \vee x_1x_2\overline{x_3x_4}$. Построяваме таблица

(4)	(3)	(2)
$(*)x_1x_2x_3x_4$	$(*)x_1x_2x_3$	x_1x_2
$(*)x_1x_2x_3\overline{x_4}$	$(*)x_1x_2x_4$	
$(*)x_1x_2\overline{x_3}x_4$	$x_1x_3x_4$	
$(*)x_1\overline{x_2}x_3x_4$	$(*)x_1x_2\overline{x_4}$	
$(*)x_1x_2\overline{x_3x_4}$	$(*)x_1x_2\overline{x_3}$	

Следователно $x_1x_3x_4 \vee x_1x_2$ е СвДНФ на нашата функция.

8. Формални езици

Определение 1 - *Азбука* ще наричаме всяко крайно множество $|V| = n$ от елементи, които ще наричаме *букви*.

Азбуките ще означаваме с главни латински или гръцки букви, а символите в дадена азбука с малки или големи букви. Тъй като азбуките са крайни множества, то по естествен начин към тях могат да се прилагат различни операции, затова ще говорим за обединение и сечени на две азбуки.
Определение 2 - Всяка крайна редица от символи от дадена азбука ще наричаме *дума (или низ)* над тази азбука. Броят на членовете на тази редица ще наричаме дължина на думата α и ще бележим $d(\alpha)$.
Определение 3 - Дума, чиято редица от символи не съдържа нито един член ще наричаме *празна дума* и ще означаваме с ϵ. Очевидно $d(\epsilon) = 0$.
Определение 4 - Две думи се наричат *равни*, когато имат една и съща дължина и съответно еднакви първи букви, еднакви втори букви, ..., еднакви последни букви, т.е. $a_1a_2 \dots a_n = b_1b_2 \dots b_m$, ако $n = m$ и $a_i = b_i$, $i = 1, \dots, n$.
Пример: Да разгледаме следните азбуки $V_1 = \{0, 1\}$, $V_2 = \{$**if**, a, b, c, **else**, (,), {, }, ;$\}$.
ϵ, 1101000 и 00 са думи над V_1 с дължини 0, 7 и 2.
if(a){}; и **if**(b){a}**else**{c}; са думи над V_2 с дължини 7 и 12.

Операции с думи:
Определение 5 - *Конкатенация (съединяване, умножение)* на думата α с β ще наричаме непосредственото записване на думата β след думата α и ще означаваме с $\alpha\beta$. Ако $\alpha = a_1a_2 \dots a_k$, $\beta = b_1b_2 \dots b_m$, то $\alpha\beta = a_1 \dots a_kb_1 \dots b_m$.

Очевидно, ако α е дума над азбуката V, а β е дума над азбуката W, то $\alpha\beta$ е дума над азбуката $V \cup W$, а $d(\alpha\beta) = d(\alpha) + d(\beta)$. За всеки три произволни думи α, β, γ при конкатенацията на α с $\beta\gamma$ се получава същата дума, както при конкатенацията на $\alpha\beta$ с γ. Това означава, че конкатенацията е асоциативна операция. Тя обаче не е комутативна операция, тъй като лесно могат да се намерят думи α и β, за които $\alpha\beta$ не съвпада с $\beta\alpha$. Освен това $\epsilon\alpha = \alpha\epsilon = \alpha$ за всяка дума α, т.е. за операцията конкатенация празната дума ϵ играе ролята на единицата при умножението. Чрез операцията конкатенация можем да дефинираме индуктивно степенуването на думи.
Определение 6 - Нека α е произволна дума. Определяме степените на α по следния начин: $\alpha^0 = \epsilon, \alpha^1 = \alpha$, а за $n = 2, 3, \dots$, $\alpha^n = \alpha^{n-1}\alpha$. Обръщане на произволна дума α ще наричаме думата $O(\alpha)$, записана в обратен ред, т.е. ако $\alpha = a_1a_2 \dots a_k$, то $O(\alpha) = a_k \dots a_2a_1$. Ще казваме, че думата α е *начало (префикс)* на думата β, ако съществува такава дума γ, че $\beta = \alpha\gamma$. Думата α ще наричаме *край (суфикс)* на думата β, ако съществува дума γ, за която $\beta = \gamma\alpha$. Думата α ще наричаме поддума на думата β, ако съществуват думи μ и η, за които $\beta = \mu\alpha\eta$.

Нека е дадена произволна азбука W. Множеството на всички думи върху азбуката W ще означаваме с W^*. Ще отбележим, че множеството W^* е затворено относно операцията конкатенация, т. е. конкатенацията на кои да са две думи от W^* е отново дума от W^*. Нека V е произволна непразна азбука. Дефинираме индуктивно следната номерация Nom на думите от V^*. Номерът на ϵ е 0, т.е. $\mathrm{Nom}(\epsilon) = 0$, а номерът на буквите от V са различни числа от 1 до n, където $|V| = n$ е броят на буквите във V. За произволна дума $\omega a \in V^*$, където $\omega \in V^*$, $a \in V$:

$$\mathrm{Nom}(\omega a) = n\mathrm{Nom}(\omega) + \mathrm{Nom}(a).$$

Лесно може да се докаже чрез индукция, че всяка дума от V^* получава точно един номер, неравните думи получават различни номера, а на всеки номер съответствува една единствена дума.

За множеството $V = \{0, 1\}^*$, получаваме-следната номерация на думите: $\mathrm{Nom}(\epsilon) = 0$, $\mathrm{Nom}(0) = 1$, $\mathrm{Nom}(1) = 2$, $\mathrm{Nom}(00) = 3$, $\mathrm{Nom}(01) = 4$, $\mathrm{Nom}(10) = 5$, $\mathrm{Nom}(11) = 6$.
Определение 7 - *Формален език* над дадена азбука ще наричаме всяко множество от думи над тази азбука. Или, с други думи, едно множество L е формален език тогава и само тогава, когато съществува такава азбука V_L, че $L \subseteq V_L$.

Пример: Да вземем азбуката $V = \{0, 1\}$. Формални езици тази азбука са например следните множества: а) празното множество $\emptyset$; б) множеството $\{\epsilon\}$ съставено от празната дума; в) V_1^*; г) $L_1 = \{0, 00, 010, 0110\}$; д) $V_2 = \{0^n1^n | n \geq 0\}$.

Операции над езици:

1) обединение $L_1 \cup L_2 = \{w \mid \omega \in L_1$ или $\omega \in L_2\}$;

2) сечение $L_1 \cap L_2 = \{w \mid \omega \in L_1$ и $\omega \in L_2\}$;

3) разлика $L_1 \setminus L_2 = \{w \mid \omega \in L_1\ \omega \notin L_2\}$;

4) допълнение $\overline{L} = V^* \setminus L = \{w \mid \omega \in V^*, \omega \notin L\}$;

5) произведение:

Определение 8 - Нека са дадени два езика: L_1 над азбуката W_1, L_2 над W_2. *Произведение* $L_1.L_2$ на езиците L_1 и L_2 ще наричаме езика

$$L_1.L_2 = \{\alpha\beta \mid \alpha \in L_1, \beta \in L_2\}$$

над азбуката $W_1 \cup W_2$, състоящ се от всички конкатенации на думи от L_1 с думи от L_2.

Пример: Нека $L_1 = \{0, 01\}$, $L_2 = \{1, 11\}$, тогава $L_1L_2 = \{01, 011, 0111\}$.

Тъй като конкатенацията на думи е асоциативна, но не е комутативна операция, непосредствено лесно се показва, че и произведението на езици е асоциативно, но не е комутативно, т. е. за произволни езици L_1, L_2, L_3 имаме

$$L_1(L_2L_3) = (L_1L_2)L_3,$$

но съществуват езици L_1 и L_2, за които $L_1L_2 \neq L_2L_1$. За тази операция езикът $\emptyset$ ролята на нула, а езикът $\{\epsilon\}$-ролята на единица, тъй като за произволен език $L.\emptyset = \emptyset.L = \emptyset$, $L\{\epsilon\} = \{\epsilon\}L = L$.

Определение 9 - Нека L е произволен формален език. Степените на L се определят но следния начин: $L^0 = \epsilon$, $L^1 = L$, а за $n = 2, \ldots ...$ $L^n = L^{n-1}L$

Вижда се, че езикът L^n се състои от всевъзможните конкатенации на n на брой думи от езика L. Следователно степените на L са също езици над азбуката V на езика L, тъй като множеството V^* е затворено относно операцията конкатенация.

Определение 10 - Итерация L^* на произволен език L ще наричаме обединението на всички степени на L:

$$L^* = \bigcup_{n \geq 0} L^n$$

Пример - а) Нека $L = \{0, 01\}$. Тогава $L^2 = \{00, 001, 010, 0101\}$, $L^3 = \{000,\ 0001,\ 0010,\ 00101,\ 01001, 01010, 010101\}$;

б) Нека $L = \{0, 1\}$. Тогава $L^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, \ldots\}$;

в) Нека $L = \{111\}$. Тогава $L^* = \{(111)^n \mid n \neq 0\} = \{\epsilon, 111, 111111, \ldots\}$.

Определение 11 - Позитивни итерация на произволен език L ще наричаме езика $L^+ = \bigcup_{n \geq 1} L^n = L^* \setminus \{\epsilon\}$.

Задачи:

1. Нека $V = \{0\}$, $W = \{1\}$. Опишете кои думи съставят следните езици: а) $(V \cup W)^*$; б) V^*W^*; в) $(VW)^*$; г) $(V \cup W)^*VW$.

2. Един език над азбуката V наричаме комутативен, ако за всеки две думи α, β от L е изпълнено $\alpha\beta = \beta\alpha$. Докажете, че L е комутативен език тогава и само тогава, когато съществува такава дума $\omega \in V^*$, че $L \subseteq \omega^*$.

9. Пораждащи граматики

Пораждаща граматика се състои от четири компоненти:

1. *азбука на терминалните символи* или *азбука на пораждания език*;

2. *азбука на нетерминалните символи*, които имат помощна роля в процеса на пораждане и не участват в думи на пораждания език, нетерминалните символи играят ролята на синтактични категории в езика;

3. *начален символ*, който представлява фиксиран нетерминален символ и съответства на началното състояние на генератора; началният символ означава общата синтактична категория на всички думи от пораждания чрез граматиката език;

4. правила на пораждащата граматика, които са краен брой и представляват наредени двойки от думи, съставени от терминални и нетерминални символи, при това в първата компонента на всяка двойка участвува поне един нетерминален символ; правилата на пораждащата граматика описват всеки процес на пораждане (генериране) на изходната терминална дума от началния символ. Пораждащата граматика поражда (или генерира) думите от езика по следния начин: започва от дума, съставена само от началния символ, и прилага правилата, като в получаваните думи замества поддума, съставена от първа компонента, с дума, представляваща втора компонента, докато се получи дума от терминални символи, Тази дума е елемент от езика.

Определение 1 - *Пораждаща граматика* Γ се нарича наредената четворка $\Gamma = \langle V, W, S, P\rangle$ в която: V е азбука на терминалните символи (терминална азбука); W е такава непразна азбука на нетерминалните символи (нетерминална азбука), че $V \cap W = \emptyset$. $S \in W$ и се нарича начален символ; P е крайно множество от наредени двойки $< \alpha, \beta >$ от думи над азбуката $V \cup W$, при което в α има поне един нетерминален символ. Елементите на P се наричат правила на Γ. Прието е правилата на граматиката да се записват във формата $\alpha \to \beta$ което се чете "думата α се замества с думата β. Знакът $\to$ не принадлежи нито на V, нито на W. Думата α се нарича лява страна на правилото, а думата β – дясна страна.

Определение 2 - Думата μ се *извежда непосредствено* от думата η в граматиката $\Gamma = \langle V, W, S, P\rangle$, ако съществуват думи $\gamma_1 \in (V \cup W)^*$ и $\gamma_2 \in (V \cup W)^*$ и правило $\alpha \to \beta$ от P такива, че: $\eta = \gamma_1\alpha\gamma_2$, $\eta = \gamma_1\beta\gamma_2$. Непосредствения извод означаваме $\eta \overset{\Gamma}{\vdash} \mu$

Определение 3 - Редица от думи $\omega_1, \omega_2, \ldots, \omega_n$, за която $\omega_1 \overset{\Gamma}{\vdash} \omega_2 \overset{\Gamma}{\vdash} \ldots \overset{\Gamma}{\vdash} \omega_n$ се нарича извод на ω_n от ω_1 в граматиката Γ. Когато съществува извод на ω_n от ω_1 в Γ, ще казваме, че ω_n се извежда от ω_1, в граматиката Γ, и ще записваме това така: $\omega_1 \overset{\Gamma}{\models} \omega_n$. Броят на непосредствените изводи, съставляващи даден извод ще наричаме *дължина* на извода.

Определение 4 - Думата α *се извежда* от граматиката $\Gamma = \langle V, W, S, P\rangle$, ако $\alpha \in V^*$ и съществува извод $S \models \alpha$. Множеството от всички думи, които една граматика $\Gamma = \langle V, W, S, P\rangle$ може да породи ще наричаме формален език породен от Γ и бележим $L(\Gamma) = \{\omega \mid \omega \in V^*, S \models \omega\}$. Две граматики Γ_1 и Γ_2 ще наричаме *еквивалентни*, ако $L(\Gamma_1) = L(\Gamma_2)$, т.е. когато пораждат един и същ език.

Пример: В граматиката $\Gamma_1 = \langle < \{a, b\}, \{S\}, S, \{S \to aSb|ab\}\rangle$ терминални символи са a и b и има само един нетерминален символ S, който е и началенТази граматика поражда например думата a^4b^4 чрез следния извод $S \to aSb \to aaSbb \to aaaSbbb \to aaaabbbb$. В този извод два пъти последователно е приложено първото правило, а след това второто. Оттук лесно можем да забележим, че граматиката Γ_1 поражда всички думи от вида a^nb^n, $n \geq 0$. Достатъчно е да приложим $n-1$ пъти първото правило към S, а след това един път второто правило. Граматиката Γ_1, поражда думи само от вида a^nb^n, $n \geq 0$. Действително първото правило запазва броя на буквите S в думите от извода, а второто правило ги намалява с една. В такъв случай при извод с начало S в думите на извода има точно по една буква S, ако се прилага първото правило,а при прилагането на второто правило буквата S изчезва и остават само терминални символи. Следователно при пораждане на думата от $L(\Gamma_1)$ второто правило може да се прилага само един път, и то в края на извода. И така единственият ред, в който могат да се прилагат правилата на Γ_1 при пораждане на дума от езика $L(\Gamma_1)$, е следният: някакъв брой пъти прилагане на първото правило, а след това един път второто правило. Но както вече видяхме, по този начин се пораждат само думи от вида a^nb^n, $n \geq 0$. Получихме, че $L(\Gamma_1) = \{a^nb^n, n \geq 0\}$.

Пример: Да разгледаме граматиката $\Gamma_2 = \langle\{0,1,2,3,4,5,6,7,8,9\},\{S\},S,\{S \to S0|S1|S2|S3|S4|S5|S6|S7|S8|S9|0|1|2|3|4|5|6|7|8|9\}\rangle$. Тази граматика поражда десетичните записвания на естествените числа 1, 2, 3, ..., 10, 11, ... Изводът на 7920 от S изглежда така $S \to S0 \to S20 \to S920 \to 7920$.

Лема 1. *Нека $\Gamma = \langle V, W, S, P\rangle$ е пораждаща граматика, съдържаща правила, в които началният символ S се среща в дясната им страна. Тогава може да се построи друга граматика Γ', еквивалентна на Γ, такава, че в десните страни на правилата на Γ' да не се среща началният и символ.*

Доказателство. Нека $S' \notin V \cup W$. Определяме Γ' по следния начин: $\Gamma = \langle V, W \cup \{S'\}, S', P'\rangle$ като P'съдържа всички правила от P и нови правила $S' \to \alpha$ за всяко правило $S \to \alpha$ от P. Очевидно S' не се среща в дясната страна на никое правило от P'.

Нека $\omega \in L(\Gamma)$. Тогава съществува извод $S \overset{\Gamma}{\models} \omega$. Нека първото правило, което се прилага в този извод, е $S \to \alpha$. В такъв случай изводът изглежда така $S \overset{\Gamma}{\vdash} \overset{\Gamma}{\models} \omega$. Съгласно дефиницията на P' в него има правило $S' \to \alpha$. Освен това P' съдържа всички правила от P. Следователно изводът $\alpha \overset{\Gamma}{\models} \omega$ може да се извърши и в граматиката Γ'. Получаваме

$$S' \overset{\Gamma'}{\vdash} \alpha \overset{\Gamma'}{\models} \omega,$$

т.е. $\omega \in L(\Gamma')$.

Нека сега $\omega \in L(\Gamma)$. Тогава $\alpha \overset{\Gamma}{\models} \omega$. Нека първото приложено в този извод правило е $S' \to \alpha$. Но тогава S' не се среща в α, в P има $S \to \alpha$, а изводът изглежда така: $S' \overset{\Gamma'}{\vdash} \alpha \overset{\Gamma'}{\models} \omega$. Тъй като S' не се среща в дясната страна на никое правило от P' и не участва в ω, то S' не се среща и в никоя от думите на извода $\alpha \overset{\Gamma'}{\models} \omega$. Но в такъв случаи всички прилагани в този извод правила не съдържат S', т.е. те са правила и от P. Получаваме, че този извод е и в граматиката Γ'. Заедно с правилото $S \to \alpha$ от P това дава извода $S \overset{\Gamma}{\vdash} \alpha \overset{\Gamma}{\models} \omega$, т.е. $\omega \in L(\Gamma)$. Следователно $L(\Gamma) = L(\Gamma')$. □

Пример: За граматиката $\Gamma = \langle < \{a,b\},\{S\},S,\{S \to aSb|ab\}\rangle$, от по-предния пример получаваме следната граматика Γ, която поражда същия език и никое правило на която не съдържа началния символ в дясната страна. $\Gamma' = \langle < \{a,b\},\{S,S'\},S',\{S \to aSb|ab, S' \to aSb|ab\}\rangle$.

10. Йерархия на Чомски

Определение 1 - Пораждащи граматики, върху правилата на които не се налагат никакви допълнителни условия, т.е. имат вида $\alpha A\beta \to \omega, \alpha, \beta, \omega \in (V \cup W)^*$, $A \in W$ се наричат *граматики от общ вид* или още от *тип 0*.

За следващия вид граматики ще искаме правилата да имат вида $\alpha A\beta \to \alpha\omega\beta$, при което $\alpha, \beta, \omega \in (V \cup W)^*$, $\omega \neq \epsilon$, а A е нетерминален символ. При такова правило A се замества с думата ω само в даден контекст $\alpha - \beta$. Прието е думите α и β da се наричат *ляв и десен контекст* на A. Чрез тези правила обаче не може да се породи празната дума, тъй като дясната страна на всяко правило не е по-къса от лявата, която от своя страна не е празна. За да могат граматики с такива правила да пораждат и празната дума, допълнително ще прибавяме правилото $S \to \epsilon$, където S е началният символ на граматиката, при условие, че S не се среща отдясно на никое правило. Тогава при пораждането на думите чрез граматиката, правилото $S \to \epsilon$ не може да се прилага в никакъв друг извод освен в извода $S \vdash \epsilon$.

Определение 2 - Пораждаща граматика $\Gamma = \langle V, W, S, P\rangle$, всички правила на която имат вида $\alpha A\beta \to \alpha\omega\beta$ се нарича *контекстна* или още от *тип 1*. Контекстна е и всяка граматика Γ, в която има едно правило $S \to \epsilon$, а всички останали правила са от горния вид, като при това S не се среща в десните им страни.

Пример: Граматиката $\Gamma = \langle\{a, b, c\}, \{S, A, B, C\}, S, \{S \to aSAC|abC, CA \to CB, CB \to BC, BC \to AC, bA \to bb, C \to c\}\rangle$ е контекстна, като в третото правило C има десен контекст A,в четвъртото правило A има ляв контекст B ...

Тази граматика поражда всички думи от вида $a^nb^nc^n$, $n \geq 1$. Действително прилагаме първото правило $n-1$ пъти към S след това второто и получаваме $S \models A^{n-1}S(AC)^{n-1} \vdash a^nbC(AC)^{n-1}$. Третото, четвъртото и петото правило в тази последователност позволяват да придвижваме буквите C през A. Получаваме $S \models a^nbA^{n-1}C^n$. След това прилагаме шестото правило $n-1$ пъти, седмото правило n пъти и получаваме $S \models a^nb^nc^n$, $n \geq 1$. Като се използва фактът, че правилата могат да се прилагат само в определен ред, може да се докаже, че Γ поражда само думи от вида $a^nb^nc^n$, $n \geq 1$, т.е. $L(\Gamma) = \{a^nb^nc^n \mid n \geq 1\}$.

Определение 3 - Пораждаща граматика $\Gamma = \langle V, W, S, P\rangle$, всички правила на която имат вида $A \to \omega$ ($A \in W$, $\omega \in (V \cup W)^+$)се нарича *безконтекстна* или още *граматика от тип 2*. Безкоктекстна е и всяка граматика Γ, в която има едно правило $S \to \epsilon$, а всички останали правила са от горния вид, като при това S не се среща в десните им страни.

Пример: Граматиката $\Gamma_2 = \langle\{0, 1\}, \{S\}, S, \{S \to SS|0S1|10|01\}\rangle$ безконтекстна. Лесно се вижда, че граматиката Γ_2 поражда само непразни думи с равен брой 0 и 1, тъй като всяко правило или не прибавя или прибавя равен брой 0 и 1 към непосредствено извежданата дума.

Определение 4 - Пораждаща граматика $\Gamma = \langle V, W, S, P\rangle$, чиито правила са от вида $A \to aB|a$, $A, B \in W$, $a \in V$, се нарича *автоматна* или още граматика от тип 3. Автоматна е и всяка ,граматика Γ, в които има едно правило $S \to \epsilon$, а всички останали са от горкия вид, като това те не съдържат S в десните си страни.

Пример Автоматна е и граматиката $\Gamma_3 = \langle\{1\}, \{S, A, B\}, S, \{S \to \epsilon|1A, A \to 1B|1, B \to 1A\}\rangle$. Вижда се, че $L(\Gamma) = \{1^{2n} \mid n \geq 0\}$. Действително всяко пораждане непразна дума в Γ трябва да завършва с четвъртото правило, и нетерминалният символ A се появява при пораждането или веднага след прилагане на първото правило, или след прилагане на първото правило последвано от неколкократно прилагане на второто и третото правило, едно след друго. И в двата случая получаваме четен брои единици.

Определение 5 - Един формален език се нарича автоматен (от тип 3), безконтекстен (от тип 2), контексен (от тип 1) или от общ вид (от тип 0) тогава и само тогава, когато има пораждаща го граматика, която е съответно автоматна (от тип 3), безконтекстна (от тип 2), контекстна (от тип 1) или от общ вид (от тип 0).

Определение 6 - Автоматните, безконтекстните, контекстните граматики и граматиките от общ вид образуват *йерархията на Чомски за пораждащите граматики*. Автоматните, безконтекстните, контекстните езици и езиците от общ вид образуват *йерархията на Чомски за формалните езици*.

От вида на правилата се вижда, че всяка автоматна граматика е безконтекстна, всяка безконтекстна граматика е контекстна, а всяка контекстна граматика може да се разглежда и като граматика от общ вид. Получаваме следната редица от включвания

автоматни $\subset$ безконтекстни $\subset$ контекстни $\subset$ общ вид

която важи както за пораждащи граматики, така и за формални езици. Докато включванията очевидно са строги за граматиките, остава да се докаже, че това е така и за езиците.

Задачи: 1. Определете езика, който се поражда от граматиката:

а) $\Gamma = \langle\{0,1\}, \{S, A, B\}, S, \{S \to \epsilon | A, A \to 0B|0A|1, B \to 0B|1\}\rangle$;

б) $\Gamma = \langle\{a, b\}, \{S, A\}, S, \{S \to \epsilon | A, A \to Ab|aA|a|b\}\rangle$.

2. Определете граматика, която поражда следния език:

а) $\{0^m 1^n \mid n \geq m \geq 1\}$;

б) $\{\alpha\alpha \mid \alpha \in \{0,1\}^*\}$.

3. Постройте автоматна граматика, пораждаща всички думи от 0 и 1, които:

а) имат нечетен брой единици;

б) не завършват с две единици;

11. Свойства на автоматните езици

Теорема 29. *Нека L и L' са произволни автоматни езици. Тогава езикът $L' \cup L''$ също е автоматен.*

Доказателство. Нека L' и L'' са автоматни езици, тогава съществуват автоматни граматики Γ' и Γ'', такива че $L = L(\Gamma')$, $L = L(\Gamma'')$. Нека $\Gamma' = \langle V', W', S', P' \rangle$, $\Gamma'' = \langle V'', W'', S'', P'' \rangle$. Ще считаме, че $W' \cap W'' = \emptyset$, $(W' \cup W'') \cap (V' \sup V'') = \emptyset$. Нека $S \notin W' \cup W'' \cup V' \cup V''$. Образуваме граматиката

$$\Gamma = \langle V' \cup V'', W' \cup W'', S, P \rangle$$

, в която P се състои от всички правила от P' и P'' с изключение на правилата $S' \to \epsilon$, $S'' \to \epsilon$ (ако има такива), от нови правила $S \to \alpha$, добавени за всяко правило $S' \to \alpha$ от P' и $S'' \to \alpha$ от P'', и от правило $S \to \epsilon$, ако в P' или P'' има правило $S' \to \epsilon$ или $S'' \to \epsilon$. Граматиката Γ е автоматна, тъй като видът на новите правила съответствува на вида на правилата от P' и P'', а S не се среща отдясно на никое правило. Ще покажем, че $L(\Gamma) = L(\Gamma') \cup L(\Gamma'')$.

Нека $\omega \in L(\Gamma)$. Ако $\omega = \epsilon$, в Γ има правило $S \to \epsilon$. Но такова правило може да има в Γ само при условие, че в Γ' или Γ'' има подобно правило. Тогава ϵ принадлежи на $L(\Gamma')$ или $L(\Gamma'')$ и следователно $\omega \in L(\Gamma') \cup L(\Gamma'')$. Да разгледаме сега случая, когато $\omega \neq \epsilon$. Тогава съществува извод в Γ:

$$S \overset{\Gamma}{\vdash} \alpha \ldots \overset{\Gamma}{\vdash} \omega, \omega \neq \epsilon, \alpha \neq \epsilon.$$

Оттук следва, че в Γ има правило $S \to \alpha \neq \epsilon$, а това е изпълнено само ако в Γ' или Γ'' има правило $S' \to \alpha$ или $S'' \to \alpha$. Да приемем, че в Γ'' има правило $S'' \to \alpha$. (Случаят, когато в Γ' има правило $S' \to \alpha$, се разглежда аналогично.) Тогава $\alpha \in (V'' \cup W'')^+$ и към α могат да се прилагат само правила от Γ"(тъй като $W' \cap W'' = \emptyset$), т. е. изводът $\alpha \models \omega$ е в граматиката Γ''. Като използваме и правилото $S'' \to \alpha$, получаваме

$$S'' \overset{\Gamma''}{\vdash} \alpha \overset{\Gamma''}{\vdash} \omega$$

В такъв случай $\omega \in L(\Gamma'')$ и следователно $\omega \in L(\Gamma') \cup L(\Gamma'')$.

Нека сега $\omega \in L(\Gamma') \cup L(\Gamma'')$. Да приемем за удобство, че $\omega \in L(\Gamma')$. Ако $\omega = \epsilon$, в Γ' има правило $S' \to \epsilon$, но тогава в Γ ще има правило $S \to \epsilon$, т. е. $\epsilon \in L(\Gamma)$. Ако $\omega \neq \epsilon$, в граматиката Γ' има извод $S \overset{\Gamma'}{\vdash} \alpha \ldots \overset{\Gamma'}{\vdash} \omega, \omega \neq \epsilon, \alpha \neq \epsilon$. Получаваме, че в Γ' има правило $S' \to \alpha, \alpha \neq \epsilon$. Но тогава в граматиката Γ ще има правило $S \to \alpha$. Освен това от $\alpha \overset{\Gamma'}{\vdash} \omega$ получаваме $\alpha \overset{\Gamma}{\vdash} \omega$, тъй като всички правила на Γ' (без $S' \to \epsilon$) са правила и на Γ. Оттук следва, че

$$S \overset{\Gamma}{\vdash} \alpha \ldots \overset{\Gamma}{\models} \omega \in L(\Gamma)$$

□

Теорема 30. *Нека L и L' са произволни автоматни езици. Тогава езикът $L'L''$ също е автоматен.*

Доказателство. Нека $\Gamma' = \langle V', W', S', P' \rangle$, $\Gamma'' = \langle V'', W'', S'', P'' \rangle$. Ще считаме, че $W' \cap W'' = \emptyset$, $(W' \cup W'') \cap (V' \sup V'') = \emptyset$. Да разгледаме отначало случая, когато L' и L'' не съдържат празната дума. Образуваме граматиката

$$\Gamma = \langle V' \cup V'', W' \cup W'', ', P \rangle$$

в които P се състои от всички правила P' и P'' с изключение на правилата в P', които имат вида $A' \to a$, $A' \to W'$, $a \in V'$ и от нови правила $A' \to aS''$ за всяко правило от P', което има вида $A' \to a$, $A' \to W'$, $a \in V'$. Граматиката Γ е автоматна, тъй като всички нови правила имат изисквания вид.

Нека $\beta \in L'L''$, т. е. $\beta = \omega'\omega''$, $\omega' \in L(\Gamma')$, $\omega'' \in L(\Gamma'')$. Това означава, че $S \overset{\Gamma'}{\models} \omega'$ и $S \overset{\Gamma''}{\models} \omega''$. Да разгледаме извода $S' \overset{\Gamma'}{\models} \omega'$. В него правило от вида $A' \to a$ се прилага само еднократно, и то в края на извода, т.е. $S' \overset{\Gamma'}{\models} \omega_1 A' \overset{\Gamma'}{\vdash} \omega_1 a = \omega'$.

Правилата, прилагани в $S' \overset{\Gamma'}{\models} \omega'_A$, са и в P, а вместо правилото $A' \to a$ от P в P е правилото $A' \to aS''$. Оттук получаваме $S' \overset{\Gamma'}{\models} \omega_1 A' \models \omega_1 a S'' = w' S''$. Като използваме извода $S'' \overset{\Gamma''}{\models} \omega''$ и това, че всички правила от P са правила и от P'', получаваме

$$S' \overset{\Gamma'}{\models} \omega' S'' \overset{\Gamma}{\models} \omega' \omega'' = \beta \in L(\Gamma).$$

Обратно, нека $\beta \in L(\Gamma)$. Това означава, че $S' \overset{\Gamma}{\models} \beta$. Към S може да се приложи правило от вида $S' \to aS''$ или правило от вида $S; \to aA'$, $a \in V'$, $A' \in W'$. Във втория случай към aA' може да се приложи правило от вида $A' \to bS''$ или правило от вида $A' \to bB'$, т.н. С други думи, в началото на извода $S' \overset{\Gamma}{\models} \beta$ се прилагат правила от вида $A' \to aB'$, докато се приложи правило от вида $A' \to aS''$, т.е. $S' \overset{\Gamma}{\models} \omega' S'' \overset{\Gamma}{\models} \omega' \omega'' = \beta$. Тъй като правилата $A' \to bB'$ са и от P', а на правилото $A' \to aS''$ от P съответства правилото $A' \to a$ от P' , то $S' \overset{\Gamma'}{\models} \omega'$. Лесно се вижда, че всички прилагани правила в извода $\omega' S'' \overset{\Gamma}{\models} \omega' \omega'' = \beta$ са правила от P , т. е. $S'' \overset{\Gamma''}{\models} \omega''$. Следователно $\beta \in L(\Gamma')L(\Gamma'')$. Тогава $L(\Gamma) = L(\Gamma')L(\Gamma'')$.

Лесно се доказва твърдението и ако единият от езиците съдържа празната дума. □

Пример: Да вземем автоматните езици $L' = \{0^n \mid n \geq 1\}$ и $L'' = \{1^m \mid m \geq 1\}$, които се пораждат от граматиките

$$\Gamma' = \langle \{0\}, \{S'\}, S', \{S' \to 0S' | 0\} \rangle,$$
$$\Gamma'' = \langle \{1\}, \{S''\}, S'', \{S'' \to 1S'' | 1\} \rangle.$$

Тяхното произведение $L'L'' = \{0^n 1^m \mid n \geq 1, m \geq 1\}$ също е автоматен и се поражда от граматиката

$$\Gamma = \langle \{0,1\}, \{S', S''\}, S', \{S' \to 0S' | 0S'', S'' \to 1S'' | 1\} \rangle.$$

Теорема 31. *Нека L е автоматен език. Тогава езикът L^* също е автоматен.*

Доказателство. Нека автоматната граматика $\Gamma = \langle V', W', S', P' \rangle$ поражда езика L. Образуваме граматиката $\Gamma = \langle V', W', ', P \rangle$, в която P се състои от всички правила на Γ с изключение на $S' \to \epsilon$ (ако има такова в Γ), а за всяко правило от вида $A \to a$, $A \in W'$, $a \in V'$ е добавено ново правило $A \to aS'$. От вида на новите правила следва, че и граматиката Γ е автоматна. Всеки път, когато в Γ' чрез правилото $A \to a$ приключва пораждането на някаква терминална дума, новата автоматна граматика Γ може или да приключи извода на думата чрез същото правило, или да приложи правилото $A \to aS'$ и с това да постави начало на извеждане на нова дума от Γ', долепена отдясно на изведената. Като повтаря това произволен брой пъти, граматиката Γ може да породи конкатенациите на произволен брой непразни думи от L. А това означава, че тя поражда езика $L^* \setminus \{\epsilon\}$. Следователно $L^* \setminus \{\epsilon\}$ е автоматен език. Но тогава обединението L^* на езика с автоматния език $\{\epsilon\}$ е също автоматен език. □

Теорема 32. *Нека L е краен език. Тогава L е автоматен.*

Пример: $L = \{001, 11\}$. Да намерим автоматна граматика, която да го разпознава.
$\Gamma = \langle \{0,1\}, \{S, A, B, C, D, E\}, S, \{S \to 0A | 1D, A \to 0B, B \to 1, D \to 1\} \rangle$.
задачи: 1. Намерете автоматна граматика, която поражда:
а) всички думи от $\{0,1\}$, които съдържат точно една нула или нечетен брой единици;
б) всички думи от $\{0,1\}$, започващи с две нули и завършващи на две единици.
задачи: 2. Намерете автоматна граматика, която поражда:
а) всички думи от $\{(ab)^n (ba)^m \mid n \geq 1, m \geq 1\}$;
б) всички думи от $\{a^k b^l c^n \mid k, l, n \geq 0\}$.

12. Детерминирани крайни автомати

Определение 1 - *Детерминиран краен автомат (ДКА)* над азбуката V наричаме наредената петорка $A = \langle K, V, \delta, q_0, F\rangle$, в която: K е непразно крайно множество от вътрешни състояния, наречено *азбука на вътрешните състояния*, V – крайно множество от входни символи, наречено *входна азбука*; δ - *функция на преходите* с дефиниционна област $D(\delta) \subseteq K \times V$ и с област на стойностите $R(\delta) \subset K$, $q_0 \in K$ – *начално състояние*; $F \subseteq K$ – множество на заключителните състояния. Детерминираният краен автомат A се нарича *напълно определен* в случай че функцията на преходите δ е дефинирана за всички наредени двойки от $K \times V$, т.е. когато $D(\delta) = K \times V$. Работата на детерминирания краен автомат A се определя по следния начин: Нека на A е зададена входната дума $a_{i_1}a_{i_2}\ldots a_{i_{k+1}} \in V^*$. По текущото състояние q_0 и първия входен символ a_{i_1} чрез функцията на преходите се определя следващото вътрешно състояние $q_1 = \delta(q_0, a_{i_1})$. По състоянието q_1 и следващия входен символ a_{i_2} чрез функцията на преходите се определя следващото вътрешно състояние и т.н. Накрая по състоянието q_k и входен символ $a_{i_{k+1}}$ се определя последното вътрешно състояние q_{k+1}. Думата се *разпознава* от автомата, ако $q_{k+1} \in F$, т.е. ако последното достигнато вътрешно състояние е заключително. В противен случай, казваме че думата *не се разпознава* от автомата A.
Определение 2 - Множеството $T(A)$ от всички думи над входната азбука V, които детерминираният краен автомат разпознава, се нарича *език разпознаван от* A.
Определение 3 - Два детерминирани крайни автомата A_1 и A_2, се наричат *еквивалентни* тогава и само тогава, когато $T(A_1) = T(A_2)$, т.е. когато разпознават един и същ език.

Функцията на преходите δ на произволен детерминиран краен автомат $A = \langle K, V, \delta, q_0, F\rangle$ е с дефиниционна област в $K \times V$. Удобно е да я додефинираме в $K \times V^*$, като се въведе функцията δ':

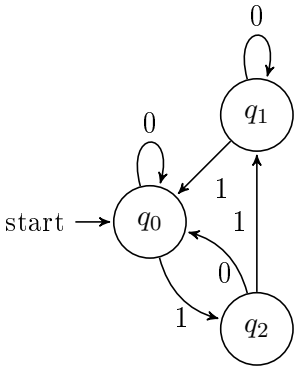
$\delta'(q, \epsilon) = q$ за всяко $q \in K$ (ϵ е празната дума); $\delta'(q, \omega a) = \delta(\delta'(q, \omega), a)$ за всяко $q \in K$, за всяка дума $\omega \in V^*$ за всяка буква $a \in V$.

В съответствие с казаното езикът $T(A)$, разпознаван от A, може да се опише така:

$$T(A) = \{\omega \mid \omega \in V^*, \delta(q_0, \omega) \in F\}.$$

Определение 4 - Диаграмата на преходите на детерминиран краен автомат A се нарича ориентираният граф с отбелязани ребра, който се получава, като за всяко вътрешно състояние на A поставим по един връх, а два върха p и q свързваме с ориентирано ребро от p към q само когато $\delta(p, a) = q$ за някое $a \in V$. Това ребро отбелязваме с буквата a. Върховете, които съответстват на заключителните състояния на A, са означени допълнително (например с двоен кръг). Допълнително е означен и върхът, съответстващ на началното състояние (обикновено с входна стрелка).

Пример: Да разгледаме детерминирания краен автомат $A = \langle\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\}, \rangle$ с функция на преходите, дефинирана по следният начин $\delta(q_0, 0) = q_0$, $\delta(q_0, 1) = q_2$ $\delta(q_1, 0) = q_1$, $\delta(q_1, 1) = q_0$ $\delta(q_2, 0) = q_0$, $\delta(q_2, 1) = q_1$ Вижда се, че A е напълно определен автомат. Ето диаграмата на този автомат.



За входната дума 011011 получаваме следната поредица от вътрешни състояния на A:

$$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_2 \in F,$$

което показва, че A разпознава думата 011011, за 1101 имаме

$$q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \notin F,$$

т.е. 1101 не се разпознава. Автоматът A може да попадне от състояние q_0 в единственото заключителното състояние q_2, ако срещне две последователни единици, и остава в това състояние, ако среща единици. Прочитането на 0 в кое да е състояние го препраща в началното състояние q_0. Оттук следва, че A разпознава всички думи от нули и единици, които завършват поне с две последователни единици, т.е.

$$T(A) = \{\omega \mid \omega \in \{0,1\}^* \text{ и } \omega \text{ не завършва с } 11\}.$$

Пример: Да разгледаме детерминирания краен автомат $A = \langle\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_0, q_1\},\rangle$ с функция на преходите, дефинирана със следната таблица:

	a	b
q_0	q_1	q_1
q_1	q_2	q_1
q_2	$\emptyset$	q_2

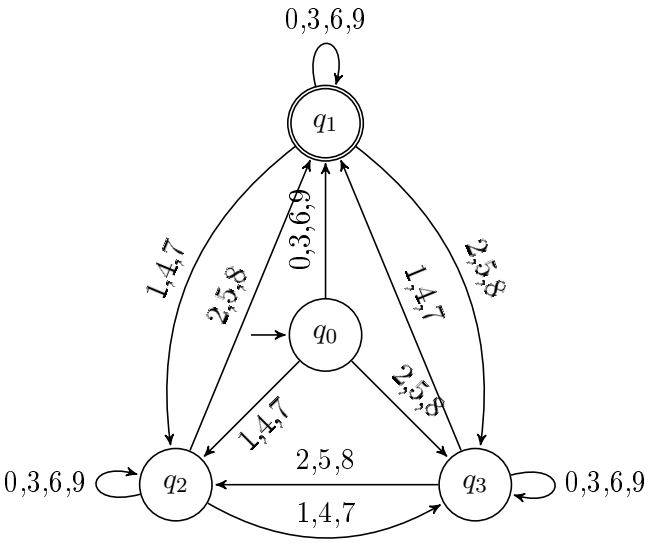
A не е напълно определен, тъй като $\delta(q_2, a)$ не е дефинирана. Оттук следва, че A не разпознава думи, започващи с b. Вижда се, че A разпознава празната дума и всички думи, които започват с буквата a и не завършват с, буквата b. Следователно автоматът A разпознава всички думи, съставени от a и b, които не започват и не завършват с буквата B.

Ще опишем начин по който за всеки детерминиран краен автомат A, който не е напълно определен, може да се построи еквивалентен на него напълно определен детерминиран краен автомат B. Автомата B получаваме, като към вътрешните състояния на A добавим ново вътрешно състояние S, което не е заключително, а към функцията на преходите на A прибавим равенства $\delta(q, a) = S$ за всяка двойка (q, a), за която автоматът A не е определен. Освен това дефинираме $\delta(S, a) = S$ за всеки входен символ a на A. Така построеният автомат B е вече напълно определен. Върху всички входни думи, за които A е определен, автоматът B действа по същия начин, както и A, т. е. B разпознава всички думи, които разпознава и A. При работа върху думите, за които A не е определен, автоматът B попада в състоянието S и остава в това състояние до края на входната дума. Тъй като S не е заключително състояние, B не разпознава тези думи. И така $T(B) = T(A)$.

Пример: За детерминирания краен автомат A от предния пример който не напълно определен получаваме следния еквивалентен на него напълно определен $B = \langle\{q_0, q_1, q_2, s\}, \{a, b\}, \delta, q_0, \{q_0, q_1\}\rangle$

	a	b
q_0	q_1	q_1
q_1	q_2	q_1
q_2	S	q_2
S	S	S

Пример: Да построим детерминиран краен автомат A, разпознаващ числата (в десетичен запис), които се делят на 3. Ще изберем четири вътрешни състояния за A – начално състояние q_0, в което попадаме, когато прочетената част от входната дума се дели на 3; q_1 – състояние, в което попадаме, когато прочетената част от входната дума при деление на 3 дава остатък 1, и накрая състояние q_2, в което A попада, когато прочетената част от входната дума при деление на 3 дава остатък 2. Тъй като A трябва да разпознава числата, които се делят на 3, то q_0 е заключително състояние. В съответствие с признака за делимост на 3 получаваме диаграмата на преходите за A.



Лема 2. *Нека е даден ДКА $A = \langle K, V, \delta, q_0, F\rangle$. Тогава за всяка дума $\alpha_1\alpha_2 \in V^*$ и за всяко вътрешно състояние $q \in K$, за които $\delta(q, \alpha_1\alpha_2)$ е дефинирано*

$$\delta(q, \alpha_1\alpha_2) = \delta(\delta(q, \alpha_1), \alpha_2).$$

Доказателство. Твърдението ще докажем чрез индукция по дължината на думата α_2. При $d(\alpha_2) = 0$ твърдението е очевидно. Нека $d(\alpha_2) = 1$, т.е. $\alpha_2 = a$, $a \in V$. Тогава съгласно начина на дефиниране на функцията δ върху $K \times V^*$ имаме $\delta(q, \alpha_1 a) = \delta(\delta(q, \alpha_1), a)$. Да допуснем, че твърдението е вярно за думите с дължина, по-малка от n. Нека α_2 е дума с дължина n. Можем да считаме, че $n > 1$. Тогава $\alpha_2 = \alpha_3 a$, $d(\alpha_3) < n$. Получаваме

$$\delta(q, \alpha_1\alpha_2) = \delta(q, \alpha_1\alpha_3 a) = \delta(\delta(q, \alpha_1), \alpha_3 a)) = \delta(\delta(\delta(q, \alpha_1), \alpha_3), a),$$

тъй като $\delta(q, \alpha_1\alpha_3) = \delta(\delta(q, \alpha_1), \alpha_3)$ според индуктивната хипотеза. Но

$$\delta(\delta(\delta(q, \alpha_1), \alpha_3), a) = \delta(\delta(q, \alpha_1), \alpha_3 a) = \delta(\delta(q, \alpha_1), \alpha_2).$$

Следователно $\delta(q, \alpha_1\alpha_2) = \delta(\delta(q, \alpha_1), \alpha_2)$. □

Теорема 33 (uvw теорема)**.** *Нека L е формален език, разпознаван от детерминиран краен автомат. Тогава съществува такива константа n, че ако α е дума от L с $d \geq n$, то α може да се представи като конкатенация на три думи u, v, w: $\alpha = uvw$, при което $d(uv) \leq n$, $d(v) \geq 1 u$ за всяко $i = 0, 1, 2, \ldots$ думите $uv^i w$ са също от езика L.*

Доказателство. Нека L се разпознава от детерминирания краен автомат $A = \langle K, V, \delta, q_0, F\rangle$ и нека в K има n вътрешни състояния. Да вземем произволна дума $\alpha = a_1 a_2 \ldots a_m$ от L, за която $d(\alpha) = m \geq n$. Да разгледаме работата на A за тази дума. Получаваме следната поредица от вътрешни състояния на A:

$$\delta(q_0, a_1) = q_{i_1}, \delta(q_0, a_1 a_2) = q_{i_2}, \ldots, \delta(q_0, a_1 a_2 \ldots a_m) = q_{i_m}.$$

Но $m \geq n$, а вътрешните състояния на A са n на брой. Тогава в редицата $q_0, q_{i_1}, \ldots, q_{i_m}$ има поне две еднакви състояния, Да вземем онези две еднакви състояния q_{i_j} и q_{i_k} $(j < k)$, които са най-наляво в редицата. Да означим думите $u = a_1 \ldots a_{i_j}$, $v = a_{i_{j+1}} \ldots a_{i_k}$, $w = a_{i_{k+1}} \ldots a_m$. Очевидно $d(uv) \leq n$, $d(v) \geq 1$, $\alpha = uvw$, $\delta(q_0, uv) = q_{i_k} = q_{i_j} = \delta(q_0, u)$ Използвайки последното равенство и предходната лема, получаваме

$$\delta(q_0, u) = \delta(q_0, uv) = \delta(\delta(q_0, u), v) = \delta(\delta(q_0, uv), v) = \delta(q_0, uv^2),$$

$$\delta(q_0, u) = \delta(q_0, uv^2) = \delta(\delta(q_0, u), v^2) = \delta(\delta(q_0, uv), v^2) = \delta(q_0, uv^3),$$

и т.н. Оттук $\delta(q_0, u) = \delta(q_0, uv) = \delta(q_0, uv^2) = \cdots = \delta(q_0, uv^i) = \ldots$ Следователно

$$\delta(q_0, uvw) = \delta(\delta(q_0, uv), w) = \delta(\delta(q_0, u), w) = \delta(q_0, uw),$$

$$\delta(q_0, uvw) = \delta(\delta(q_0, uv), w) = \delta(\delta(q_0, uv^i), w) = \delta(q_0, uv^iw),$$

за $i = 1, 2, \ldots$, т. е. за всяко i думите uv^iw принадлежат на $T(A)$. Допълнително получихме, че константата от условието на теоремата може да се приеме равна на броя на вътрешните състояния на детерминирания краен автомат, разпознаващ L. □

Следствие 3. *Нека L е език, разпознаван от детерминиран краен автомат A с n състояния. L не е празен език тогава и само тогава, когато A разпознава думи с дължина, по-малка от n.*

Следствие 4. *Съществува алгоритъм, който определя дали един език, разпознаван от детерминиран краен автомат, е празен, или не.*

Следствие 5. *Съществува безконтекстен език, който не се разпознава от детерминиран краен автомат.*

Доказателство. Ще докажем с помощта на теоремата, че за безконтекстния език $L(\Gamma) = \{a^nb^n \mid n \geq 1\}$, породен от граматиката $\Gamma\langle\{a, b\}, \{S\}, S, \{S \to aSb|ab\}\rangle$, не съществува детерминиран краен автомат, който да го разпознава. Да допуснем, че езика се разпознава от някакъв детерминиран краен автомат A. Тогава за достатъчно голямо n, като приложим uwv-теоремата, получаваме, че a^nb^n може да се представи във вида uvw, $d(v) \geq 1$ и за всяко $i = 0, 1, 2, \ldots$ думите uv^iw са от $L(\Gamma)$. Ако $v \in \{a, b\}^+$ и $v \notin \{a\}^+$, $v \notin \{b\}^+$, то $uv^2w \notin L(\Gamma)$, тъй като за думите от $L(\Gamma)$ всички букви a се намират пред буквите b. Следователно $v \in \{a\}^+$ или $v \in \{b\}^+$. Ако $v \in \{a\}^+$ или $v \in \{b\}^+$, то $uw \notin L(\Gamma)$, тъй като думите от $L(\Gamma)$ имат равен брой букви a и b, а в думата uw ще има по-малко букви от този вид, които съставят v. Следователно $v \notin \{a\}^+$, $v \notin \{b\}^+$. Получаваме противоречие. В такъв случай за безконтекстния език $L(\Gamma)$ не съществува детерминиран краен автомат, който да го разпознава. □

задачи: Постройте ДКА, който да разпознава:
а) всички думи от $\{0, 1\}^*$, които завършват точно с три последователни единици;
б) всички думи от $\{0, 1\}^*$, за които 010 е поддума;
в) всички думи от $\{0, 1\}^+$, които не започват с две последователни нули;
г) всички думи от $\{0, 1\}^*$, които започват с някаква степен на думата 101.

13. Недетерминирани крайни автомати. Връзка между крайни автомати и автоматни езици.

Определение 1 - *Недетерминиран краен автомат (НДКА)* над азбуката V наричаме наредената петорка $A = \langle K, V, \delta, q_0, F\rangle$, в която: K е непразно крайно множество от вътрешни състояния, наречено *азбука на вътрешните състояния*, V – крайно множество от входни символи, наречено *входна азбука*; δ - *функция на преходите* с дефиниционна област $D(\delta) \subseteq K \times V$ и с област на стойностите $R(\delta) \subseteq P^K$ (множеството от всички подмножества на K), $q_0 \in K$ – *начално състояние*; $F \subseteq K$ – множество на заключителните състояния.

По текущото състояние q_0 и първия входен символ a_{i_1} чрез функцията на преходите се определя множеството от следващите вътрешни състояния $\delta(q_0, a_{i_1}) = \{q_1, \ldots, q_l\}$. По всяко от тези състояния $q_1, \ldots, q_l$ и следващия входен символ a_{i_2} чрез функцията на преходите се определя следващото множество от вътрешни състояния и т.н. Накрая по множеството от състояния $q_{k_1}, \ldots, q_{k_s}$ и последния входен символ $a_{i_{k+1}}$ се определя последното множество от вътрешни състояния $q_{l_1}, \ldots, q_{l_t}$. Думата се *разпознава* от автомата, ако поне едно от състоянията $q_{l_1}, \ldots, q_{l_t}$ е заключително, т.е. $\{q_{l_1}, \ldots, q_{l_t}\} \cap F \neq \emptyset$. В противен случай $\{q_{l_1}, \ldots, q_{l_t}\} \cap F = \emptyset$, казваме че думата *не се разпознава* от автомата A.

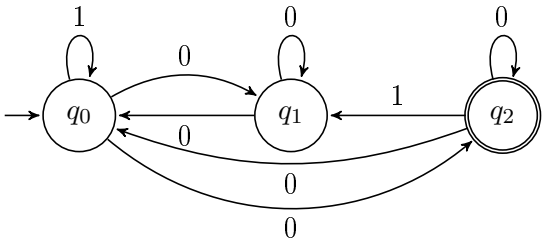
Определение 2 - Множеството $T(A)$ от всички думи над входната азбука V, които детерминираният краен автомат разпознава, се нарича *език разпознаван от* A.

$$T(A) = \{\omega \mid \omega \in V^*, \delta(q_0, \omega) \cap F \neq \emptyset\}.$$

Пример: Даден е НДКА $A = \langle\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\}\rangle$ с функция на преходите

	0	1
q_0	$\{q_1, q_2\}$	$\emptyset$
q_1	$\{q_0\}$	$\{q_0, q_2\}$
q_2	$\{q_0, q_1\}$	$\{q_2\}$

Върху думата 01000 автомата действа така: $\delta(q_0, 0) = \{q_1, q_2\}$ $\delta(q_0, 01) = \delta(q_1, 1) \cup \delta(q_2, 1) = \{q_1\}$ $\delta(q_0, 010) = \delta(q_1, 0) = \{q_0, q_1\}$ $\delta(q_0, 0100) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1, q_2\}$ $\delta(q_0, 01000) = \{q_1, q_1, q_2\}$ и тъй като тук има заключително състояние, то думата 01000 се разпознава.



Теорема 34. *Нека L е език, който се разпознава от недетерминиран краен автомат. Тогава съществува детерминиран краен автомат, който също разпознава L.*

Доказателство. Нека $A = \langle K, V, \delta, q_0, F\rangle$ е НДКА, който разпознава L. Строим следния детерминиран краен автомат $A' = \langle Q, V, \delta', t_0, F'\rangle$ в който:

Q съдържа по едно вътрешно състояние за всяко подмножество на K. Вътрешното състояние, съответстващо на подмножеството $\{p1, \ldots, p_n \subseteq K\}$, ще означаваме с $t_{[p_1,\ldots,p_n]}$

за всяка буква a и всяко вътрешно състояние $t_{[p_1,\ldots,p_n]}$ $\delta'(t_{[p_1,\ldots,p_n]}, a) = t_{[r_1,\ldots,r_l]}$ тогава и само тогава, когато

$$\{r_1, \ldots, r_l\} = \delta(p_1, a) \cup \delta(p_2, a) \cup \cdots \cup \delta(p_n, a).$$

$t_0 = t_{[q_0]}$

F' се състои от всички вътрешни състояния $t_{[p_1,\ldots,p_n]}$, за които $\{p_1, \ldots, p_n\} \cap F \neq \emptyset$.

Ще докажем, че за произволна дума $\omega \in V^*$, $\delta'(t_0, \omega) = t_{[p_1,\ldots,p_n]}$ тогава и само тогава, когато $\delta(q_0, \omega) = \{p_1, \ldots, p_n\}$. Ще използваме индукция по дължината на думата ω. Нека $\omega = \epsilon$. Тъй като $\delta(q_0, \epsilon) = q_0$, то $\delta'(t_0, \epsilon) = \delta'(t_{[q_0]}, \epsilon) = t_{[q_0]}$ съгласно дефиницията на δ'. Обратно, ако $\delta'(t_{[q_0]}, \epsilon) = t_{[q_0]}$, то $\{q_0\} = \delta(q_0, \epsilon)$.

Нека да допуснем, че за всички думи $\omega \in V^*$, за които $d(\omega) \leq N$, $\delta'(t_0, \omega) = t_{[p_1,\ldots,p_n]}$ тогава и само тогава, когато $\delta(q_0, \omega) = \{p_1, \ldots, p_n\}$ Нека ωa е произволна дума с дължина $N+1$. За думата ωa получаваме $\delta'(t_0, \omega a) = \delta'(\delta'(t_0, \omega), a) = \delta'(t_{[p_1,\ldots,p_n]}, a)$ тогава и само тогава, когато $\delta(q_0, \omega) = \{p_1, \ldots, p_n\}$.

По дефиниция $\delta'(t_{[p_1,\ldots,p_n]}, a) = t_{[r_1,\ldots,r_l]}$ тогава и само тогава, когато $\{r_1, \ldots, r_l\} = \delta(p_1, a) \cup \delta(p_2, a) \cup \cdots \cup \delta(p_n, a)$. Следователно тогава и само тогава, когато

$$\{r_1, \ldots, r_l\} = \delta(p_1, a) \cup \delta(p_2, a) \cup \cdots \cup \delta(p_n, a) \text{ при } \{p_1, \ldots, p_n\} = \delta(q_0, \omega)$$

т. е. когато

$$\{r_1, \ldots, r_l\} = \bigcup_{p_i \in \delta(q_0,\omega)} \delta(p_i, a).$$

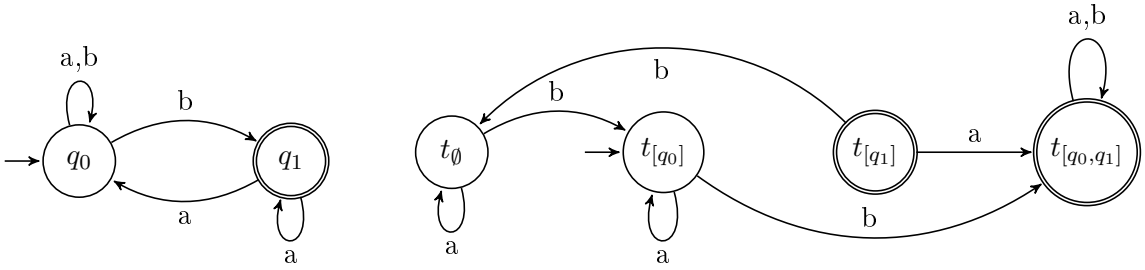
Но $\bigcup_{p_i \in \delta(q_0,\omega)} \delta(p_i, a) = \delta(q_0, \omega a)$. Следователно $\delta'(t_0, \omega a) = t_{[r_1,\ldots,r_l]}$ тогава и само тогава, когато $\{r_1, \ldots, r_l\} = \delta(q_0, \omega a)$.

Ще използваме доказаното твърдение, за да покажем, че недетерминираният краен автомат A и детерминираният краен автомат A' разпознават един и същ език. Действително $\delta'(t_0, \alpha) = t_{[p_1,\ldots,p_n]}$ тогава и само тогава, когато $\delta(q_0, \alpha) = \{p_1, \ldots, p_n\}$. Но $t_{[p_1,\ldots,p_n]}$ е заключително състояние тогава и само тогава, когато в $\{p_1, \ldots, p_n\}$ има поне едно заключително състояние. Следователно A и A' едновременно разпознават или не думата α, т. е. $T(A) = T(A')$. □

Пример: Нека е даден НДКА $A = \langle \{q_0, q_1\}, \{a, b\}, \delta, q_0, \{q_1\} \rangle$ с функция на преходите $\delta(q_0, a) = \{q_0\}$, $\delta(q_0, b) = \{q_0, q_1\}$, $\delta(q_1, a) = \{q_0, q_1\}$, $\delta(q_1, b) = \emptyset$. Да построим детерминиран краен автомат A', който също разпознава $T(A)$. Състояния на A' ще бъдат $t_\emptyset, t_{[q_0]}, t_{[q_1]}, t_{[q_0,q_1]}$, по едно за всяко подмножество на $\{q_0, q_1\}$. Функцията на преходите δ' за A се дефинира така:

	$t_\emptyset$	$t_{[q_0]}$	$t_{[q_1]}$	$t_{[q_0,q_1]}$
a	$t_\emptyset$	$t_{[q_0]}$	$t_{[q_0,q_1]}$	$t_{[q_0,q_1]}$
b	$t_\emptyset$	$t_{[q_0,q_1]}$	$t_\emptyset$	$t_{[q_0,q_1]}$

Начално състояние на A ще бъде $t_{[q_0]}$, а заключителни състояния $t_{[q_1]}$ и $t_{[q_0,q_1]}$ Диаграмата на преходите на A и A' са следните



Теорема 35. *Нека L е автоматен език. Тогава съществува недетерминиран краен автомат, който разпознава L.*

Доказателство. L е автоматен език, следователно има автоматна граматика $\Gamma = \langle V, W, S, P \rangle$, която поражда L. Строим следния недетерминиран краен автомат: $A = \langle K, V', \delta, q_0, F \rangle$ в който:

$$K = W \cup \{E\}, E \notin V \cup W, V' = V$$

$\delta(B, a)$ съдържа всички състояния D, за които в Γ има правило $B \to aD$ и състоянието E, ако в Γ има правило $B \to a$, $B, D \in W$, $a \in V$. $\delta(E, a) = \emptyset$ за всяко $a \in V$, $q_0 = S$.

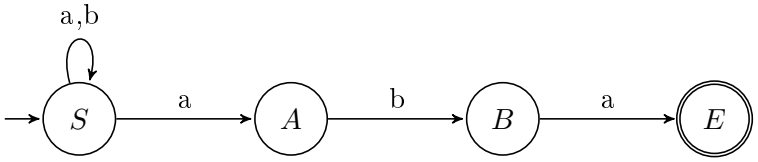
$F = \{E\}$, ако в Γ няма правило $S \to \epsilon$ и $F = \{E, S\}$, ако в Γ има правило $S \to \epsilon$. □

Пример: Нека е дадена автоматна граматика $\Gamma = \langle \{a, b\}, \{S, A, B\}, S, \{S \to aS|bS|aA, A \to bB, B \to a\} \rangle$. Получаваме следния НДКА, разпознаващ $L(\Gamma)$: $A = \langle \{S, A, B, E\}, \{a, b\}, \delta, S, \{E\} \rangle$, в

който

	S	A	B	E
a	$\{S, A\}$	$\emptyset$	E	$\emptyset$
b	S	B	$\emptyset$	$\emptyset$

с диаграма



Теорема 36. *Нека L е език, който се разпознава от детерминиран краен автомат. Тогава съществува автоматна, която също разпознава L.*

Доказателство. Нека L се разпознава от ДКА $A = \langle K, V, \delta, q_0, F\rangle$. Определяме следната автоматна граматика $\Gamma = \langle V, K, q_0, P\rangle$, където P съдържа правила $q \to ap$ за всяко равенство $\delta(q, a) = p$ и допълнителни правила $q \to a$ за всяко равенство $\delta(q, a) = p$, в което $p \in F$. Лесно може да се докаже, че $L(\Gamma) = L$. □

Оттук получаваме извода, че ДКА и НДКА разпознават езиците, породени от автоматни граматики и само тях.

задачи: 1. Постройте НДКА, който да разпознава езика:

а) $L = \{\omega 000 \mid \omega \in \{0, 1\}^*\}$;

б) $L = \{a\omega a \mid \omega \in \{a, b\}^*\}$.

2. Докажете, че допълнението на един автоматен език е също автоматен език.

14. Регулярни изрази. Теорема на Клини

Определение 1 - Нека V е азбука.

1. $\emptyset$ е *регулярен израз над* V и представлява празното множество;
2. ϵ е *регулярен израз над* V и представлява множеството $\{\epsilon\}$;
3. a е *регулярен израз над* V за всяка буква $a \in V$ и представлява множеството $\{a\}$;
4. x и y са регулярни изрази над V, представляващи множествата X и Y, то $(x+y)$, (xy) и (x^*) са също *регулярни изрази над* V, представляващи множествата $X \cup Y$, XY и X^*;
5. *регулярни* са само изразите, които могат да се получат след краен брой прилагане на правилата 1–4.

Често пъти при записване на регулярните изрази ще пропускаме скобите, когато това не нарушава еднозначността на прочитането. При това ще считаме, че итерацията е по-силна операция от умножението, а умножението – по силна операция от събирането (подобно на обичайните аритметични изрази). В такъв случай $ab^*a + ba$ означава $((a((b^*)a)) + (ba))$.

Пример: $1^* + 0(1^*0)^*11^*$ е регулярен израз и представя множеството

$$\{1\}^* \cup (\{0\}^*.\{\{1\}^*.\{0\}\}^*.\{1\}.\{1\}^*).$$

Ще отбележим, че два различни регулярни израза могат да представят едно и също множество. Например $(ba)^*b$ и $b(ab)^*$ представят множеството от всички думи от $\{a,b\}^*$, които започват и завършват с b и в които a и b се редуват.

Определение 2 - Регулярнн изрази, които представят едно и също множество, се наричат *еквивалентни*.

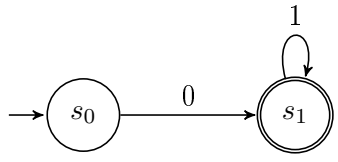
Следващите еквивалентности за произволни регулярни изрази над дадена азбука се доказват с непосредствена проверка: 1) $x + y = y + x$; 2) $(x + y) + z = (x + y) + z$; 3) $(xy)z = x(yz)$; 4) $(x + y)z = xz + yz$; 5) $\emptyset^* = \epsilon$; 6) $x + x = x$; 7) $x(y + z) = xy + xz$; 8) $(x + y)^* = (x^* + y^*)^*$; 9) $(x + y)^* = (x^*y^*)^*$; 10) $(x^*)^* = x^*$; 11) $xx^* + \epsilon = x^*$

Тези еквивалентности ни дават възможност да преобразуваме и опростяваме регулярните изрази, когато искаме да определим множеството, което представят.

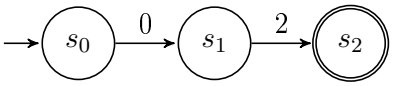
Пример: Да построим недетерминиран краен автомат, които да разпознава множеството, представяно от регулярния израз $(01^* + 02).1 + (0 + 1).(220^*1)^*$.

За целта първо ще построим недетерминираните крайни автомати:

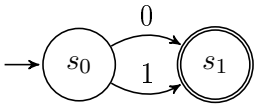
а) разпознаващ множеството, представено от 01^*;



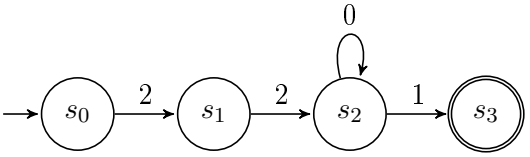
б) разпознаващ множеството, представяно от 02;



в) разпознаващ множеството, представяно от $0 + 1$;

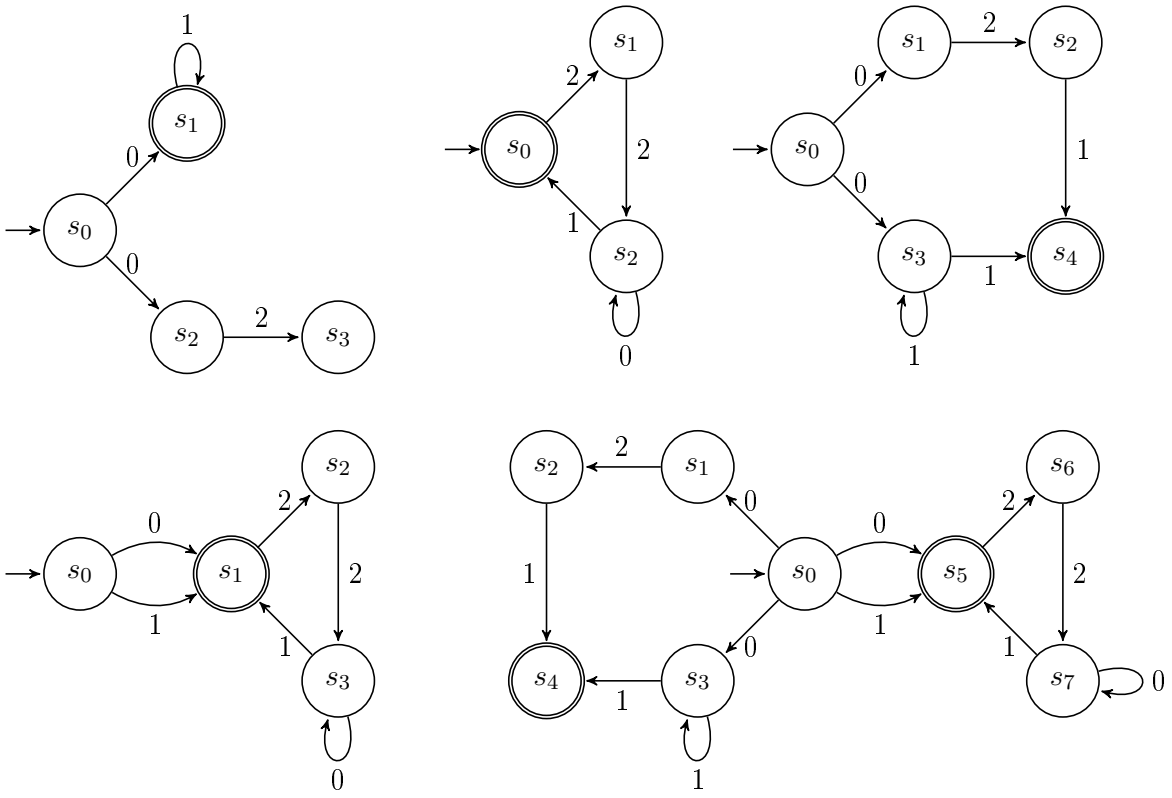


г) разпознаващ множеството, представяно от 220^*1.



Като използваме тези автомати, можем да построим НДКА:

д) разпознаващ множеството, представяно от $01^* + 02$;



е) разпознаващ множеството, представяне от $(220^*1)^*$;
ж) разпознаващ множеството, представяна от $(01^* + 02)1$;
з) разпознаващ множеството, представяно от $(0 + 1)(220^*1)^*$.

От тези автомати получаваме окончателния автомат, разпознаващ множеството

$$(01^* + 02).1 + (0 + 1).(220^*1)^*.$$

Теорема 37 (Клини)**.** *Един език $L \subseteq V^*$ е автоматен тогава и само тогава, когато се представя чрез регулярен израз над V.*

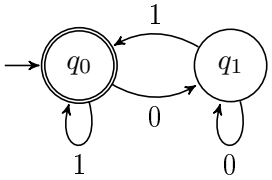
Без доказателство.

Нека да определим множествата R_{ij}^k за всяко $i, j = 1, \ldots, n$ по следния начин:

$$R_{ij}^0 = \begin{cases} \{a \mid a \in V, \delta(q_i, a) = q_j\}, & \text{ако } i \neq j \\ \{a \mid a \in V, \delta(q_i, a) = q_j\} \cup \{\epsilon\}, & \text{ако } i = j \end{cases}$$

$$R_{ij}^k = R_{ij}^{k-1} \cup R_{ik}^{k-1}(R_{kk}^{k-1})^* R_{kj}^{k-1}$$

Пример: Нека A е детерминираният краен автомат, който е зададен чрез диаграмата на преходите



Да представим езика $T(L)$ чрез регулярен израз над $\{0, 1\}$ Имаме $T(A) = R_{00}^1 = R_{00}^0 \cup R_{01}^0 (R_{11}^0)^* R_{10}^0$.

Трябва да определим регулярните изрази, които представят множествата R_{00}^0, R_{01}^0, R_{11}^0, R_{10}^0. Получаваме:

R_{00}^0 се представя от 1^*;
R_{01}^0 се представя от 1^*0
R_{11}^0 се представя от $1^*0 + \epsilon$
R_{10}^0 се представя от 11^*. Тогава $T(A)$ се представя от регулярния израз $1^* + 1^*0(1^*0 + \epsilon)^*11^*$.

задачи: 1. Докажете, че следните регулярни изрази над азбуката $\{0,1\}$ са еквивалентни: $(0+1)^*$, $0^*(10^*)^*$.

2. Проверете кои от следните еквивалентности са изпълнени за произволни регулярни изрази x, y и z над дадена азбука V:

а) $(x+y)^* = x^* + y^*$; б) $x + (x+y)^* = (x+yz)^*$;

в) $(x+y)^* = x^*(x+y)^*$; г) $(x(x+y)^* + z)^* = (x+y+z)^*$.

3. Постройте ДКА, който да разпознава, множеството означено от следния регулярен израз:

а) $bba(a+b)^*$;

б) $(a+b)^*bab$.

15. Минимизация на крайните автомати

Нека $A = \langle K, V, \delta, q_0, F\rangle$ е произволен, напълно определен детерминиран краен автомат. Думите от V^* могат да се класифицират в зависимост от крайното състояние, в което попада A при работата си над тях. Следователно автоматът A по естествен начин определя над думите от V^* следната релация R_A: две думи от V^* са свързани с релацията R_A тогава и само тогава, когато довеждат A от началното състояние q_0 до едно и също вътрешно състояние.

Определение 1 - Нека $A = \langle K, V, \delta, q_0, F\rangle$ е напълно определен детерминиран краен автомат. $\alpha R_A \beta$ тогава и само тогава, когато $\delta(q_0, \alpha) = \delta(q_0, \beta)$.

Чрез непосредствена проверка може да се установи, че релацията е рефлексивна, симетрична и транзитивна и следователно е релация на еквивалентност върху V. Всяка релация на еквивалентност разделя множеството, върху което е определена, на непресичащи се класове на еквивалентност. Тъй като автоматът A е напълно определен, релацията R_A е дефинирана върху цялото множество V^* и го разделя на непресичащи се класове от думи, като в един и същ клас попадат онези думи от $V*$, работата на A над които завършва с едно и също състояние. Оттук следва, че броят на класовете на еквивалентност, на които релацията R_A разделя V^*, не е по-голям от броя на вътрешните състояния на автомата A.

Както е известно, броят на класовете на еквивалентност се нарича индекс на релацията на еквивалентност. Получаваме следния резултат: За всеки напълно определен краен детерминиран автомат A релацията на еквивалентност R_A, която той задава върху V^*, има краен индекс.

Определение 2 - Една релация на еквивалентност R върху V^* се нарича *дясно инвариантна* (относно конкатенацията) тогава и само тогава, когато от $\alpha R \beta$ следва $\alpha\gamma R \beta\gamma$ за всяко $\gamma \in V^*$.

Лесно се вижда, че релацията R_A е дясно инвариантна

Задаването на произволен език $L \subseteq V^*$ определя върху V^* следната релация: Две думи от V^* са свързани с тази релация тогава и само тогава, когато десните им конкатенации с коя да е дума от V^* едновременно принадлежат или едновременно не принадлежат на L.

Определение 3- Нека $L \subseteq V^*$, $\alpha R_L \beta$ тогава и само тогава, когато $\alpha, \beta \in V^*$ и $\alpha\gamma \in L$ за всяко $\gamma \in V^*$ тогава и само тогава, когато $\beta\gamma \in L$.

Чрез непосредствена проверка се установява, че R_L е рефлексивна, симетрична и транзитивна и следователно е релация на еквивалентност върху V^*.

Релацията R_L е също дясно инвариантна.

Теорема 38. *Нека L е произволен формален език над азбуката V. L е автоматен език тогава и само тогава, когато релацията R_L има краен индекс.*

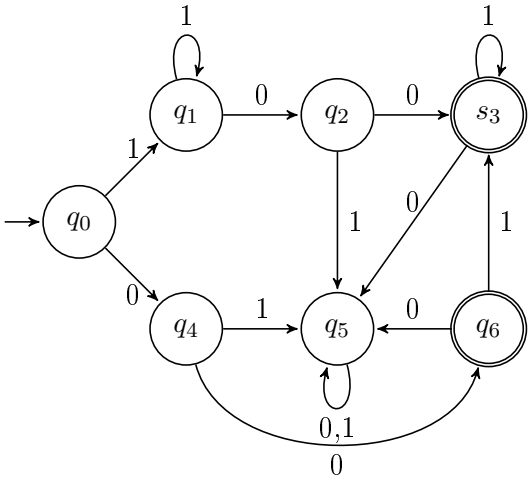
Доказателство. $\Rightarrow$) Нека L е произволен автоматен език. Тогава съществува детерминиран краен автомат $A = \langle K, V, \delta, q_0, F\rangle$,който разпознава L. Без ограничаване на общността можем да считаме, че A е напълно определен, тъй като в противен случай можем да го доопределим. Автоматът A задава върху L релацията R_A. Нека $\alpha R_A \beta$ за $\alpha, \beta \in V^*$. Релацията R_A е дясно инвариантна и следователно $\alpha\gamma R_A \alpha\gamma$, т.е. $\delta(q_0, \alpha\gamma) = \delta(q_0, \beta\gamma) =$. Ако $\alpha\gamma \in L$, то $\delta(q_0, \alpha\gamma) \in F$ и следователно $\delta(q_0, \beta\gamma) \in F$. Аналогично, ако $\alpha\gamma \notin L$, то $\beta\gamma \notin L$. Получаваме $\alpha R_L \beta$. И така за всеки две думи $\alpha, \beta \in V^*$, ако $\alpha R_A \beta$, то $\alpha R_L \beta$. Това означава, че всеки клас на еквивалентност на релацията R_L е обединение на класове на еквивалентност на R_A. Но R_A има краен индекс. Оттук следва, че и R_L също има краен индекс.

$\Leftarrow$) Нека L е произволен формален език над азбуката V, за който релацията R_L има краен индекс. Ще построим детерминиран краен автомат M, който да разпознава L. Да означим класовете на еквивалентност на R_L с $[\alpha]$ ще означаваме класа на еквивалентност, от който е думата α. Тъй като класовете не се пресичат, всеки клас се определя от кой да е негов представител. Вътрешни състояния на M ще бъдат класовете на еквивалентност на релацията R_L. Те са краен брой, тъй като R_L има краен индекс. Входна азбука на M ще бъде V. Функцията на преходите δ се дефинира по следния начин: За всяко $a \in V$ и за всяко вътрешно състояние $[\alpha]$ задаваме $\delta([\alpha], a) = [\alpha, a)]$. Релацията R_L е дясно инвариантна и в такъв случай десните конкатенации на думите от един и същ клас с произволна буква a ще бъдат отново в един и същ, възможно друг клас на еквивалентност. Ако представител на първия клас е думата α, за представител на втория клас можем да вземем

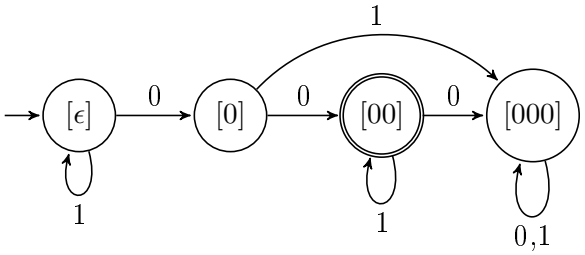
думата αa. Начално състояние на M ще бъде $[\epsilon]$, а заключителни-класовете, съставени от думи от L.

Ще покажем, че $T(M) = L$. Нека $\omega \in L$ Получаваме $\delta([\epsilon], \omega) = [\epsilon\omega] = [\omega]$. Но $[\omega]$ е заключително състояние, защото $\omega \in L$. Тогава $\omega \in T(M)$. Нека сега $\omega \in T(M)$. Следователно $\delta([\epsilon], \omega)$ е заключително състояние. В такъв случай $\delta([\epsilon], \omega) = [\epsilon\omega] = [\omega]$ е клас от думи на езика L, т.е. $\omega \in L$. Получихме, че L се разпознава от детерминирания краен автомат M. Следователно L е автоматен език. □

Пример: Нека детерминираният напълно определен краен автомат A, е зададен с диаграмата на преходите



Да определим класовете на еквивалентност на релацията R_A. Да означим с C_i, $i = 0, 1, 2, 3, 4, 5, 6$ класа от входни думи α, за които $\delta(q_0, \alpha) = q_i$. Получаваме $C_0 = \epsilon$, $C_1 = 11^*$, $C_2 = 11^*0$, $C_3 = 11^*001^*0011$, $C_4 = 0$, $C_5 = (0+1)^*(01 + 000 + 11^*01 + 11^*001^*0 + 0011^*0)$, $C_6 = 00$ като съответните множества са представени с регулярни изрази. Вижда се, че всички състояния са достижими от началното състоя- ние. Езикът $T(A))$ се състои от класовете C_3 и C_6, съответстващи на заключителните състояния, т.е. $T(A) = C_3 \cup C_6 = 0011^* + 00 + 11^*001^* = 1^*001^*$. Да определим класовете на еквивалентност на релацията R_L за $L = T(A) = 1^*001^*$. Класовете на еквивалентност са $[\epsilon]$, [0], [00] и [000] и получаваме автомата



Теорема 39. *Нека $L \subseteq V^*$ е произволен автоматен език. Тогава сред ДКА, разпознаващи L съществува единствен (с точност до преименуване на вътрешните състояния) автомат с минимален брой вътрешни състояния.*

Доказателство. По релацията R_L построяваме детерминирания краен автомат M от предходната теорема $M = \langle K, V, \delta, q_0, F\rangle$. Нека M е произволен ДКА, разпознаващ L. Броят на вътрешните състояния на A е по-голям или равен на броя на класовете на еквивалентност на R_A. От друга страна, класовете на еквивалентност на R_L са обединение на класове на еквивалентност на R_A, т.е. броят на класовете на еквивалентност на R_A е по-голям или равен на броя на класовете на R_L. Оттук получаваме, че броят на състоянията на произволен автомат A, разпознаващ L, е по-голям или равен на броя на състоянията на M, тъй като класовете на R_L са състоянията на M.

Единственост няма да доказваме. □

Нека е даден произволен напълно определен детерминиран краен автомат $A = \langle K, V, \delta, q_0, F\rangle$, разпознаващ езика L. Тук ще изложим един прост метод за намиране на минималния краен ав-

томат M, разпознаващ същия език L. Състояния на минималния краен автомат M са класовете на еквивалентност на релацията R_L. Класовете на еквивалентност на R_L са съставени от класове на еквивалентност на релацията R_A. Да наречем едно вътрешно състояние q на A достижимо, ако съществува входна дума ω, за която $\delta(q_0, \omega) = q$. Тогава между класовете на еквивалентност на R_A и достижимите състояния на A има взаимно еднозначно съответствие. Да наречем еквивалентна онези достижими състояния, чиито съответни класове на еквивалентност на релацията R_A образуват един клас на еквивалентност на релацията R_L. Това означава, че достижимите състояния p и q са *еквивалентни* $p \equiv q$ тогава и само тогава, когато $\delta(p, \omega)$ и $\delta(q, \omega)$ за всяка входна дума ω едновременно са или не са заключителни състояния. Две състояния p и q наричаме *различни*, ако $p \not\equiv q$, т.е. ако съществува дума ω, за която $\delta(p, \omega)$ и $\delta(q, \omega)$ не принадлежат едновременно на F или на $K \setminus F$. И така, за да се намери минималният краен автомат M, съответстващ на дадения автомат A, трябва да се изключат всички недостижими състояния и да се намерят класовете от еквивалентни достижими състояния. Ще използваме следната процедура за намиране на различимите състояния на автомата A:

1. Маркираме всички двойки $\{p, q\}$ от състояния на A, за които $p \in F$, а $q \in K \setminus F$.

2. За всяка немаркирана двойка $\{p, q\}$ от състояния на A разглеждаме двойките $\{\delta(p, a), \delta(q, a)\}$ за всички букви $a \in V$. Ако някоя от тези двойки е вече маркирана, то маркираме и двойката $\{p, q\}$. Ако никоя от тези двойки не е маркирана, поставяме $\{p, q\}$ в списък, асоцииран с всяка от двойките $\{\delta(p, a), \delta(q, a)\}$.

3. Ако в процеса на на маркиране на двойките от състояния се маркира някоя двойка $\{p, q\}$, маркират се всички двойки в асоциирания с двойката $\{p, q\}$ списък.

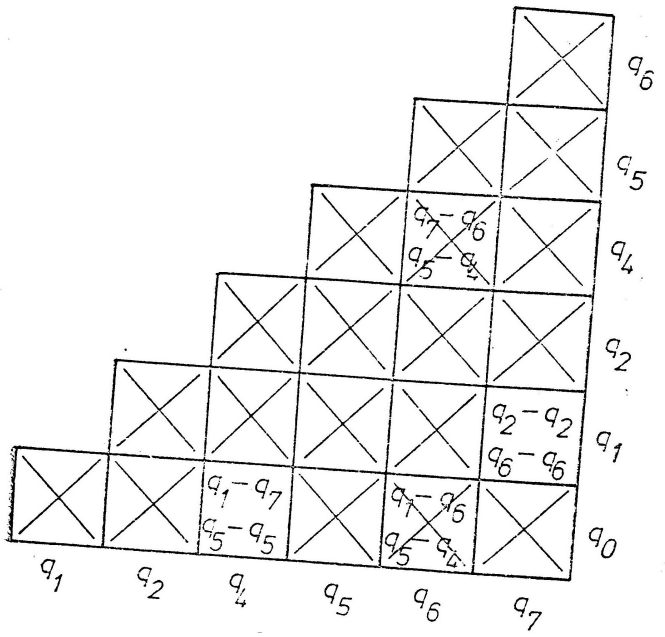
Пример: Нека е даден напълно определен ДКА

$$A = \langle \{q_0, \ldots, q_7\}, \{0, 1\}, \delta, q_0, \{q_2\} \rangle$$

	q_0	q_1	q_2	q_3	q_4	q_5	q_6	q_7
0	q_1	q_6	q_0	q_2	q_7	q_2	q_6	q_6
1	q_5	q_2	q_2	q_6	q_5	q_6	q_4	q_2

Да определим всички състояния на A, достижими от началното състояние q_0. Получаваме

$Q_0 = \{q_0\} \subset Q_1 = \{q_0, q_1, q_5\} \subset Q_2 = \{q_0, q_1, q_2, q_5, q_6\} \subset Q_3 = \{q_0, q_1, q_2, q_4, q_5, q_6\} \subset Q_4 = \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\} = Q_5 = \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}$. Следователно състоянието q_3 е единственото недостижимо състояние на A. Да определим сега различимите състояния на автомата A, като резултатите от прилагането на процедурата отбелязваме в таблица.



Прилагаме т. 1 и маркираме следните двойки: $\{q_0, q_2\}$, $\{q_1, q_2\}$,$\{q_2, q_4\}$,$\{q_2, q_5\}$,$\{q_2, q_6\}$,$\{q_2, q_7\}$. След това към немаркираните двойки прилагаме т. 2 и отбелязваме получените асоциирани с тях двойки в съответните клетки.

Получаваме следните класове на еквивалентни достижими състояния $\{q_0, q_4\}$, $\{q_1, q_7\}$, $\{q_2\}$, $\{q_5\}$, $\{q_6\}$. Минималният автомат изглежда така: