The Ministry of Education and Science of Ukraine
Dnipro University of Technology

T. Kagadiy, A. Shporta

The fundamentals of discrete mathematics

Textbook

Dnipro
Dniprotech
2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Національний
технічний університет
ДНІПРОВСЬКА
ПОЛІТЕХНІКА
1899

Т.С. Кагадій, А.Г. Шпорта

Основи дискретної математики

Навчальний посібник

Дніпро
НТУ «ДП»
2022

УДК 519.1
К 45

Затверджено вченою радою університету як навчальний посібник для здобувачів вищої освіти за спеціальностями 121 Інженерія програмного забезпечення та 122 Комп'ютерні науки (протокол № 1 від 03.01.2023)

Рецензенти:

А.Є. Шевельова – д-р фіз.-мат. наук, професор, професор кафедри обчислювальної математики та математичної кібернетики Дніпровського національного університету ім. Олеся Гончара;

В.Б. Говоруха – д-р фіз.-мат. наук, професор, зав. кафедри вищої математики, фізики та загальноінженерних дисциплін Дніпровського державного аграрно-економічного університету;

Кагадій Т.С.

К 45 The fundamentals of discrete mathematics = Основи дискретної математики: textbook / T. Kagadiy, A. Shporta; The Ministry of Education and Science of Ukraine, Dnipro University of Technology . – Dnipro: Dniprotech, 2022. – 78 p.

The textbook contains theoretical information from the main sections of the Discrete Mathematics course: set theory, relation theory, Boolean functions and transformations, graph theory. Comprehensive solutions to a large number of examples are given, options for independent solving are offered. The material is presented in detail and sequentially, which makes it possible to study it independently or through distance education.

Designed for students of higher education majoring in 121 Software Engineering and 122 Computer Science, who are studying information technology and also want to improve their English. With the help of the theoretical and practical material of the textbook, students should improve their ability to think mathematically and logically, investigate discrete mathematical models, solve applied problems, and apply discrete mathematics methods when creating a software product.

Містить теоретичні відомості з основних розділів дисципліни «Дискретна математика»: теорія множин, теорія відношень, булеві функції та перетворення, теорія графів. Наведено вичерпні розв'язки великої кількості прикладів, запропоновано варіанти для самостійного розв'язування. Матеріал викладено докладно і послідовно, що дає змогу вивчати його самостійно або при дистанційній формі навчання.

Розроблено для здобувачів вищої освіти за спеціальностями 121 Інженерія програмного забезпечення та 122 Комп'ютерні науки, які вивчають інформаційні технології, а також прагнуть покращити англійську мову. За допомогою теоретичного та практичного матеріалів даного посібника студенти мають удосконалити вміння математично та логічно мислити, досліджувати дискретні математичні моделі, розв'язувати прикладні задачі та застосовувати методи дискретної математики при створенні програмного продукту.

УДК 519.1

Contents

Introduction

The main ways of presenting information are discrete: these are words and constructions of languages and grammars - natural and formalized; tabular arrays of real data; information of social, demographic, economic statistics, etc.

Methods for processing and analyzing such discrete information are studied by discrete mathematics

The textbook (academic guide) is designed primarily for students studying mathematical methods in computer science and technology

It contains the main topics that relate to discrete mathematics (sets, relational theory, Boolean algebra, graph theory)

For better assimilation of the material, a large number of solved examples are given in the text, some of the similar tasks are offered for independent study

Examples of applied problems are given, for the solution of which methods and algorithms of discrete mathematics are used, which demonstrates the relevance and necessity of studying this subject.

The material of the textbook sections is presented in the form of lectures. It is adapted for both independent and classroom work and contains information, without which it is impossible to study the course "Discrete Mathematics". The content of each lecture is logically structured and subject to a specific topic, but not time frames.

Section 1. ***Sets. Actions on them***

The **set** is general and at the same time primary concept. Therefore, we give its definition intuitively. **A set** is a collection of elements that have a common property. The sets are indicated in capital letters and their elements in corresponding lower case:

$$\begin{aligned} &A=\{a_1,a_2,...,a_n\}; \\ &B=\{b_1,b_2,...,b_n\}; \\ &N=\{1,2,3,...\}. \end{aligned} \tag{1}$$

Elements of the set can be other sets.

$$\begin{aligned} &A=\{C,D\} \text{ or } A=\{\{c_1,c_2\},\{d_1,d_2,d_3\}\}; \\ &B=\{b_1,\{a_1,a_2\}\}. \end{aligned} \tag{2}$$

A set is called **finite** if it contains a finite number of elements and **infinite** if the number of elements is unlimited.

As a rule, the order of elements in a set does not play a role, but there are sets where such an order is taken into account. For example, the coordinates of a point

If an element is repeated in a task, in a set it is written once.

Ways to Define Sets.

1. The first way is listing the elements (1)

This method is unsuitable for infinite sets and even for finite with a large number of elements (set of fish in the Pacific Ocean).

2. The second way is to set a specific property of the elements. For example: a) natural numbers less than ten

$$N_{10}=\{x: x\in N,\ x<10\} \tag{3}$$

b) a circle is a geometrical place of points equidistant from the center.

3. The third method is recursive, when the way to get subsequent elements from previous ones is indicated. For example, Fibonacci numbers (each number is the sum of the previous two).

<u>Basic concepts of set theory.</u>

Two sets are called **equal** if they contain the same set of elements $A = B$.

The number of elements in a finite set is called its **power or cardinal number** and is denoted $|A|$.

The set *A*, all of whose elements belong to the set *B*, is called a **subset** of the set *B*.

Subordination of **non-strict order** means that *A* is a subset of *B* and can coincide with *B*, $A \subseteq B$.

A **strict order** relationship is denoted as $A \subset B$. *A* is the own subset of *B*.

A set that contains all the possible elements for a given task is called **universal** *U*.

A set that contains no elements is called **empty** $\varnothing$.

The set of all subsets of the set *A* is called the **Boolean** of the set *A* and denoted

$$2^A. \tag{4}$$

The **power** of a Boolean is calculated by the formula $|2^A| = 2^{|A|}$.

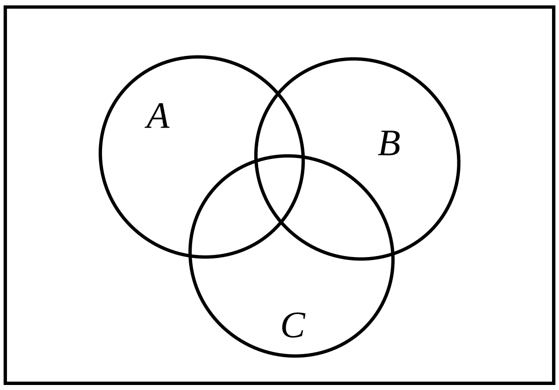
<u>Geometrical interpretation of sets</u>

To visualize the relationships between subsets of a universal set, Vienn diagrams and Euler circles are used.

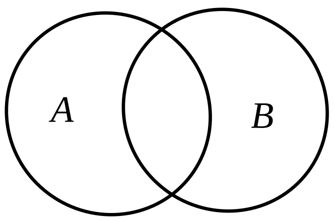
The construction of Vienne diagrams consists of dividing the plane into 2^n areas using n shapes.

Moreover, each next figure should have only one common area with each of the previously constructed figures.

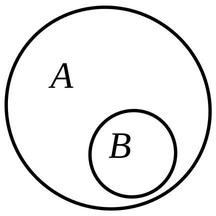
The plane on which the figures are depicted is a universal set



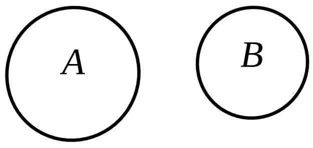
Vienn diagrams do not reflect real relationships between sets. Individual relationships between sets are represented using Euler circles.



Some *A* are not *B*



All *B* are *A*

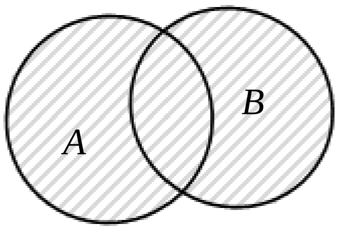


No *A* are *B*

Operations on Sets

1. A **union** (sum) is a set that consists of elements that belong to the set *A* or the set *B*

$$A \cup B = \{x : x \in A \text{ or } x \in B\}$$

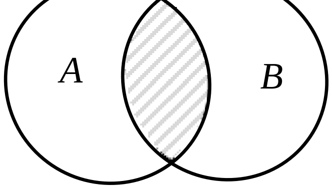


Example 1. $A = \{a,b,c\}$; B=$\{b,m,n\}$;

$A \cup B = \{a,b,c,m,n\}$;

2. An **intersection** (product) is a set that consists of elements that belong to *A* and *B* at the same time

$$A \cap B = \{x : x \in A \text{ and } x \in B\}$$



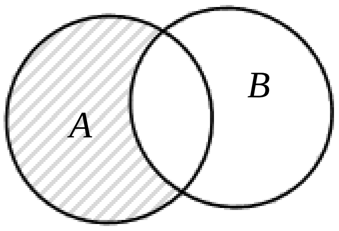
$A \cap B$

Example 2. $A=\{a,b,c\}$; B=$\{m,b,n\}$;

$A\cap B=\{b\}$;

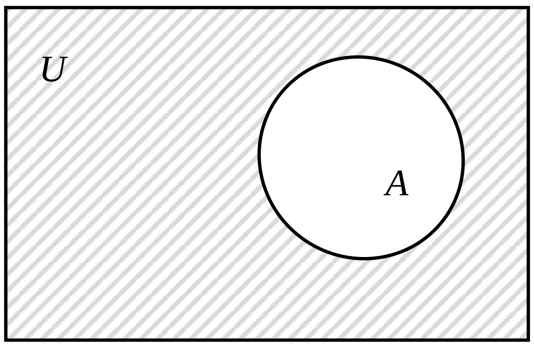
3. The **difference** of the sets consists of all elements that belong to the first set but do not belong to the second

$$A\setminus B=\{x: x\in A \text{ and } x\notin B\}$$



4. **Negation** (complement) *A* consists of elements not belonging to *A*

$$\bar{A} \text{ or } U\setminus A$$



The difference of the sets can be expressed through the negation and intersection operations

$$A\setminus B=A\cap\bar{B}$$

<u>Algebra of sets</u>

The set of all subsets of a universal set with given operations make up the algebra of sets.

Set algebra is widely used in programming, in particular, when working with databases, and also the basis for many mathematical structures.

In addition, our whole life passes among the sets that are interconnected.

When applying operations on sets, their priority should be taken into account.

1. $\overline{A}$ (negation);
2. $A \cap B$ (intersection);
3. $A \cup B$ (union);
4. $A \setminus B$ (difference).

Example 3. $E = \left(A \setminus B \cup \overline{A} \cap D\right) \setminus B;$

$$E = \left(A \overset{4}{\setminus} \left(B \overset{3}{\cup} \left(\overset{1}{\overline{A}} \right) \overset{2}{\cap} D \right) \right) \overset{5}{\setminus} B;$$

The following laws apply in set algebra

Commutative law:

$$A \cup B = B \cup A; \qquad A \cap B = B \cap A;$$

Associative law:

$$A \cup (B \cup C) = (A \cup B) \cup C; \qquad A \cap (B \cap C) = (A \cap B) \cap C;$$

Distributive law:

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C); \qquad A \cap (B \cup C) = (A \cap B) \cup (A \cap C).$$

Some properties of the universal and empty sets:

$A \cup \varnothing = A;$ $\qquad A \cap U = A;$

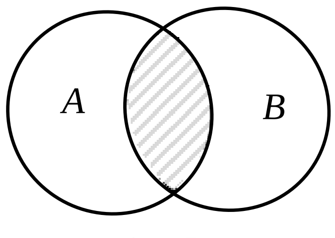
$A \cup U = U;$ $\qquad A \cap \varnothing = \varnothing;$

$A \cup \overline{A} = U;$ $\qquad A \cap \overline{A} = \varnothing;$ $\qquad \overline{\overline{A}} = A;$

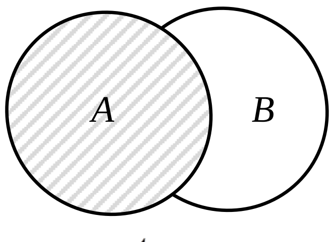
$A \cup A = A;$ $\qquad A \cap A = A.$

Law of Absorption:

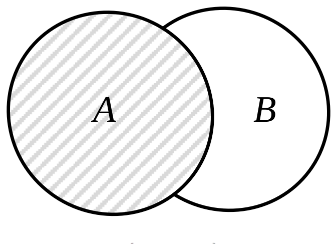
$$A \cup (A \cap B) = A; \qquad A \cap (A \cup B) = A.$$



$A \cap B$



A

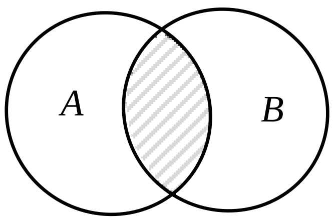


$A \cup (A \cap B)$

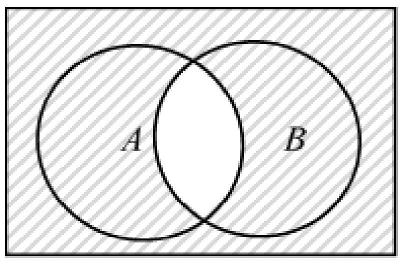
The second equality follows similarly.

De Morgan Law:

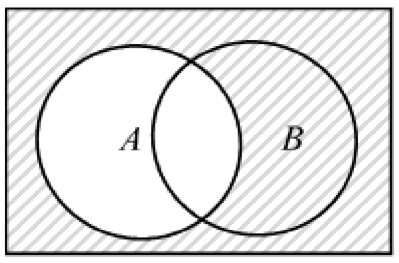
$$\overline{A\cup B}=\bar{A}\cap\bar{B};\qquad \overline{A\cap B}=\bar{A}\cup\bar{B};$$



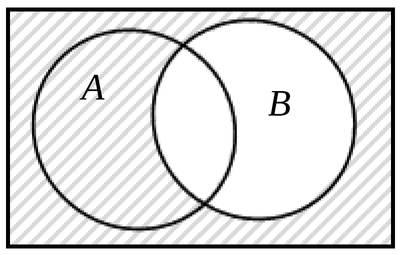
$A\cap B$



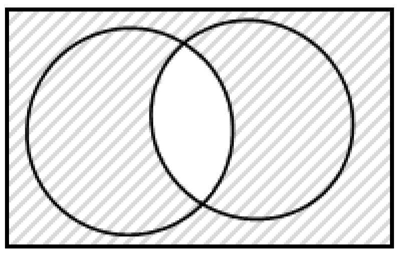
$\overline{A\cap B}$



$\bar{A}$



$\bar{B}$



$\bar{A}\cup\bar{B}$

Example 4. Simplify expression

$\overline{\left(\bar{A}\cup B\cup C\right)}\cap\left(A\cap\left(B\cup\bar{C}\right)\right)\cap\bar{B}=$ (apply de Morgan's law)

$=\left(\bar{A}\cap B\cap C\right)\cap\left(A\cap\left(B\cup\bar{C}\right)\right)\cap\bar{B}=$ (associativity and commutativity)

$=A\cap\bar{B}\cap\bar{C}\cap A\cap\bar{B}\cap\left(B\cup\bar{C}\right)=$ (apply the law of idempotence)

$=A\cap\bar{B}\cap\bar{C}\cap\left(B\cup\bar{C}\right)=$ (apply the law of distributivity)

$=A\cap\bar{B}\cap\bar{C}\cap B\cap A\cap\bar{B}\cap\bar{C}\cap\bar{C}=$ (property of an empty set)

$=\varnothing\cup A\cap\bar{B}\cap\bar{C}=A\cap\bar{B}\cap\bar{C}.$

The set must be define correctly. For example, the only hairdresser in the city defines a lot of men whom he shaves, as those who do not shave themselves. For whom is this definition incorrect?

For the hairdresser himself. He shaves himself and a hairdresser shaves him.

This contradiction is called the logical paradox.

Control questions

1. What is a set?
2. In what ways can you specify a set?
3. What are the operations on sets?
4. What are the properties of operations on sets?

<u>Section 2.</u> Relations

Relationships realize in mathematical terms on abstract sets real relationships between real objects. Relations are used when building computer databases. Using relationships, they describe the relationships between different groups of data, and also process this data. Relationships are used in programming. The constituent data structures describe some sets along with the relationships between the elements of these sets.

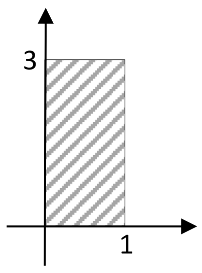
For the formal description of various combinations with elements of sets that are part of a relation, the concept of the Cartesian product of sets is used.

The Cartesian product of the sets A and B is the set of all kinds of pairs, such that the first element of the pair from the set A, and the second from the set B.

$$A\times B=\{(a,b): a\in A,\ b\in B\}; \qquad (1)$$

Example 1. $A=\{2,5,6\};\ \mathrm{B}=\{3,1,2\};$

$$A\times B=\{(2,3)(2,1)(2,2)(5,3)(5,1)(5,2)(6,3)(6,1)(6,2)\};$$



Continues interval

$A=[0,1]$; B=$[0,3]$;

The shaded area is the Cartesian product.

The Cartesian product of n sets is the set of all possible ordered sets of n elements, which are called **tuples** of length n. The first element of the **tuple** belongs to the first set, the second to the second set, and so on.

$$A_1 \times A_2 \times ... \times A_4 = \{(x_1, x_2, ..., x_4) : x_1 \in A_1,\ x_2 \in A_2, ..., x_4 \in A_4\}; \qquad (2)$$

If the same set is multiplied in a Cartesian product, n times it is called the degree of the set

$$\underbrace{A \times A \times ... \times A}_{n} = A^n; \qquad (3)$$

Example 2. $A = \{a,b,c\}$;

$A^2 = \{(a,a)(a,b)(a,c)(b,a)(b,b)(b,c)(c,a)(c,b)(c,c)\}$;

The number of elements or the power (cardinal number) of the degree of the set is determined by the formula

$$|A^n| = |A|^n; \qquad (4)$$

Example 3. $|A| = 3$; $|A^2| = 3^2 = 9$.

Cartesian product is non-commutative. The following properties are also valid

1) $A \times B \neq B \times A$;
2) $A \times (B \cup C) \neq (A \times B) \cup (A \times C)$; (5)
3) $A \times (B \setminus C) \neq (A \times B) \setminus (A \times C)$;

The power (cardinal number) of the Cartesian product is determined by the formula

$$|A \times B| = |B \times A| = |A| \cdot |B|. \qquad (6)$$

Let’s get back to Ex. (1).

The *n*-ary relation R on sets $X_1, X_2, \ldots$ is a subset of the Cartesian product of these n sets

$$R \subseteq X_1 \times \ldots \times X_n. \tag{7}$$

If n=1 we have an **unary relation**, if n=2 **binary relation**

$$(x, y) \in R \text{ or } xRy. \tag{8}$$

Binary relationships are basic. Since the Cartesian product is associative, it can be represented as a chain of binary relations.

Ways to set binary relations

1) The relation can be specified as a list of pairs of which the relation consists

$A = \{1, 2, 3, 4, 7\}$, $B = \{24, 25, 26\}$

relation: "divider", "to be a number divider", "*x* divisor *y*". (9)

$R = \{(1,24)(1,25)(1,26)(2,24)(2,26)(3,24)(4,24)\};$

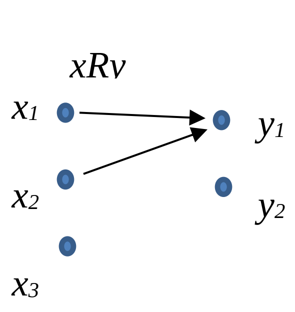
2) The second way - using the matrix

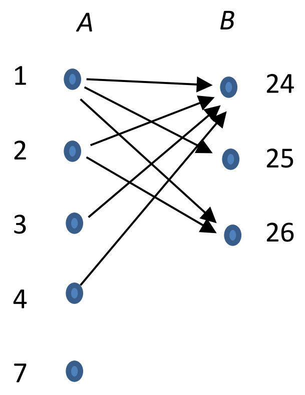
(10)

B \ *A*	24	25	26
1	1	1	1
2	1	0	1
3	1	0	0
4	1	0	0
7	0	0	0

B \ *A*	columns
rows	

3) A relationship can be graphically represented using a shape called a graph. Elements of the sets *X* and *Y* are represented on the plane by points. If the pair (*x*, *y*) belongs to the relation, the points are connected by a line directed from the first element to the second (arc). The points themselves are called the vertices of the graph.





(11)

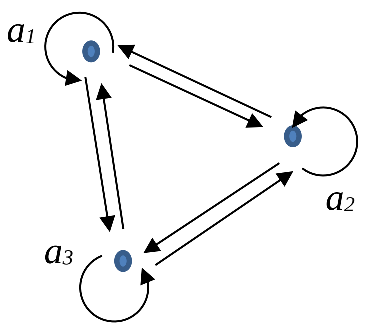
Historically, the first graph can be considered a genealogical tree. One can consider the relation of being a father, etc.

<u>Some special cases</u>

Let a binary relation be given on the set A: $R \subseteq A^2$.

1) If $R = A^2$, then the relation is called complete. What is its matrix and graph?

A \ A	a1	a2	a3
a1	1	1	1
a2	1	1	1
a3	1	1	1

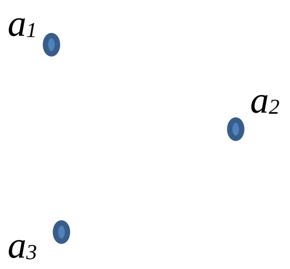


(12)

The matrix consists of units, the graph is also called complete and is shown in the figure.

2) If $R = \varnothing$ then the set is called empty.

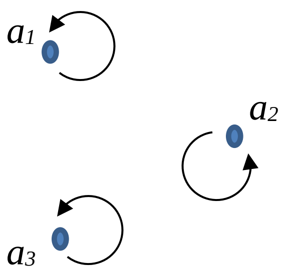
A \ A	a1	a2	a3
a1	0	0	0
a2	0	0	0
a3	0	0	0



(13)

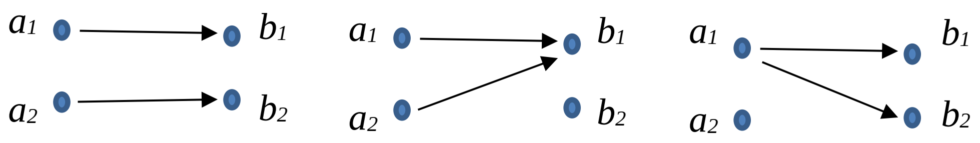
3) If the relation contains only pairs of the form (a, a) then it is called identical. In the matrix on the main diagonal are units, and the remaining elements are zeros. The graph contains only loops.

$A \backslash A$	a1	a2	a3
a1	1	0	0
a2	0	1	0
a3	0	0	1



If the pairs (x, y) make up the relation, then the ordered pairs (y, x) make up the **inverse** relationship. In the graph, the arcs are directed in the opposite direction.

The relationship between the sets X and Y is called **functional** if all its elements (ordered pairs) are different in the first element. The relation in the third figure is not functional



(14)

Properties of binary relations

1. A relation on a set X is called **reflexive** if it holds for any element with respect to itself. In the matrix of such a relation, there are units on the main diagonal. In the graph, each vertex has a loop.

2. A relationship is called **anti-reflexive** if it follows from the fulfillment of the condition $x_1 R x_2$ that $x_1 \neq x_2$. In the matrix, all diagonal elements are zero, and in the graph, there are no loops.

The relation $\leq$ on the set of real numbers is reflexive, the relation $<$ on the set of real numbers is antireflexive. The relation "to have a common divisor" on a set of integers is reflexive, the relation "to be a son" on a set of people is anti-reflective.

3. A relation is called **symmetric** if, when it is executed from the first element to the second, it is also satisfied from the second to the first.

The matrix is symmetrical with respect to the main diagonal. In the graph, all arcs have parallel ones.

4. The relation is called **asymmetric** if from the execution from the first element to the second it follows that the relation is not satisfied from the second to the first.

5. A relation is called **antisymmetric** if from the execution of it from the first element to the second and from the second to the first it follows that these elements are equal.

6. A relation is called **transitive** if it follows from fulfilling it from the first element to the second and from the second to the third that it is fulfilled from the first to the third.

7. The relation is called **antitransitive** if from the execution from the first element to the second and from the second to the third it follows that from the first to the third the relation is not fulfilled.

The relation “to be equal”, “to be a divisor” on a set of integers, as well as the relation “to live in one city” on a set of people are transitively, and the relation “to be a son” is anti-transitively.

Most commonly used are the following binary relationship classes:

1. The equivalence relation has the properties of reflexivity, symmetry, and transitivity.

2. A non-strict order relation has the properties of reflexivity, antisymmetry, and transitivity.

3. A strict order relation has the properties of antireflexivity, asymmetry, and transitivity.

4. The relation of tolerance has the properties of reflexivity, symmetry and antitransitivity.

Control questions

1. Construct a matrix and graph for the following relation defined on the set $A=\{1,2,3\}$.

a) $\{(1,1),(1,2),(1,3)\}$;

b) $\{(1,1),(2,1),(2,3),(2,3)\}$;

c) $\{(1,1),(1,2),(1,3),(2,2),(2,3),(3,3)\}$;

d) $\{(1,3),(3,1)\}$.

2. Construct a graph and list of items (elements) for relations $A=\{a,b,c\}$.

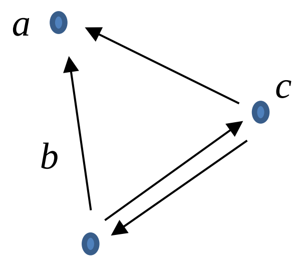
$A \backslash A$	a	b	c
a	1	0	1
b	0	1	0
c	1	0	1

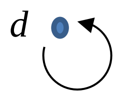
$A \backslash A$	a	b	c
a	0	1	0
b	0	1	0
c	0	1	0

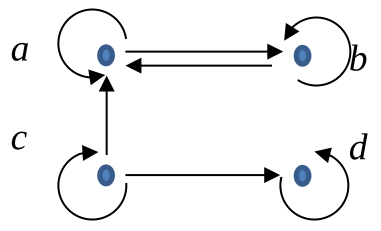
$A \backslash A$	a	b	c
a	1	1	1
b	1	0	1
c	1	1	1

$A \backslash A$	a	b	c
a	1	0	1
b	0	1	0
c	1	0	1

3. Construct a matrix and list of items for relations







4. Write a list of elements for 3-ary relation given on the set of natural numbers $(a,b,c)\in R:\ 0<a<b<c<5$.

5. Write a list of elements for 4-ary relation given on the set of natural numbers $(a,b,c,d)\in R,\ abcd=6$.

<u>Section 3.</u> Boolean functions

George Bull developed a binary logic apparatus in the middle of the 19th century. Used when working with databases, designing intelligent systems, to analyze the operation of computers and other electronic devices.

Consider the **two-element set *B*** whose elements are denoted by {1,0}.

Variables that can only take values from the set *B* are called logical or **Boolean variables**. 0 and 1 are boolean constants.

A function of the form $y = f(x_1, x_2, ...x_n)$ whose arguments and values belong to the set B are called a Boolean function.

A set of n specific values of Boolean variables $(x_1, x_2, ...x_n)$ is called a binary word or interpretation of a Boolean function.

The number of all possible Boolean functions of n variables is 2^{2^n}.

The number of values that a function takes in n variables is two to the power of n.

A variable x of an n-dimensional Boolean function is called **dummy** if

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

for any values of the remaining variables. A dummy variable can be deleted or entered. The remaining variables are called **significant.**

Example 1. The function of three variables takes values (10011110). Determine which of the variables are dummy and which are significant.

xyz	*f(x,y,z)*
000	1
001	0
010	0
011	1
100	1
101	1
110	1
111	0

1) $f(000) = f(100)$, $f(001) \neq f(101)$
   x-significant;
2) $f(000) \neq f(010)$, y-significant;
3) $f(000) \neq f(001)$, z-significant.

Ways to Define Boolean Functions

1. Using the truth table.
2. Using serial number.
3. Using formula.

Tables in which each set of variables corresponds to a function value is called a **truth table**. The set of variable values is written in ascending order as numbers in the binary system. Consider the Boolean functions of one variable (Table 1) and two variables (Table 2).

Table 1

x	φ_0	φ_1	φ_2	φ_3
0	0	0	0	1
1	0	1	0	1

$\varphi_0 = 0$ - constant 0;

$\varphi_1 = x$ - argument repeat function;

$\varphi_2 = \bar{x}$ - argument negation function;

$\varphi_3 = 1$ - constant 1.

Table 2

Function	*Designation and name*
$f_0(x,y)$	0, constant 0
$f_1(x,y)$	$x \wedge y = xy$, conjunction ("x and y")
$f_2(x,y)$	$x \setminus y$, ("$x \wedge \bar{y}$")("x and not y")
$f_3(x,y)$	x; repetition of the first argument
$f_4(x,y)$	$y \setminus x$, ("$y \wedge \bar{x}$")("y and not x")
$f_5(x,y)$	y; repetition of the second argument
$f_6(x,y)$	$x \oplus y$, "XOR", "modulo 2 sum"
$f_7(x,y)$	$x \vee y$, disjunction ("x or y")
$f_8(x,y)$	$x \downarrow y$, Pierce arrow ("not x and not y")
$f_9(x,y)$	$x \square y$, equivalence
$f_{10}(x,y)$	$\bar{y}$, "not y"
$f_{11}(x,y)$	$y \to x$, reverse implication
$f_{12}(x,y)$	$\bar{x}$, "not x"

$f_{13}(x,y)$	$x \to y$, implication
$f_{14}(x,y)$	$x \mid y$, Schaeffer stroke reverse implication, ("$\overline{x}$ or $\overline{y}$")
$f_{15}(x,y)$	1, constant 1

Operation priority: negation, conjunction, disjunction, implication, equivalence. In Boolean algebra for the basic operations there are the same laws as in set theory.

Example 2. Build a truth table for a function with a given formula

$$f(x,y,z) = x \to y \wedge z \vee \overline{x}$$

	4		*2*		*3*	*1*	
x	$\to$	y	$\wedge$	z	$\vee$	--	x
0	1	0	0	0	1	1	0
0	1	0	0	1	1	1	0
0	1	1	0	0	1	1	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	0	1
1	0	0	0	1	0	0	1
1	0	1	0	0	0	0	1
1	1	1	1	1	1	0	1

The function $f_*(x_1,...,x_n)$ is called **dual** to the function $f(x_1,...,x_n)$ if

$$f_*(x_1,...,x_n) = \overline{f}(\overline{x_1},...,\overline{x_n}).$$

A function is called **self-dual** if $f_* = f$.

In order to build a truth table of a function dual to a given one, it is necessary to change each value of the Boolean function in the truth table of this function and write the resulting column in the reverse order.

Control questions

1. What is the truth table of a function?

2. In what order are Boolean algebra operations performed?

3. What is a dummy variable?

4. On what set are Boolean variables considered?

Section 4. ***Disjunctive decompositions of Boolean functions***

Among the set of equivalent formulas that represent the selected Boolean function, one formula stands out, which is called the perfect normal form of the function.

This form has a regulated logical structure and is uniquely determined by the type of function. The construction of this form is based on the recurrent application of theorems on special decomposition of Boolean functions in variables. We consider some of these theorems.

To simplify the mathematical calculations, we introduce the binary parameter σ and the notation x^{σ} as follows: $x,\ \sigma \in B = \{0;1\}$,

$$x^{\sigma} = \begin{cases} \overline{x}, \text{ if } \sigma = 0, \\ x, \text{ if } \sigma = 1. \end{cases}$$

We can conclude that

$$x^{\sigma} = \begin{cases} 1, \text{ if } x = \sigma, \\ 0, \text{ if } x \neq \sigma. \end{cases}$$

It is directly verified that the binary function $f(x,\sigma) = x^{\sigma}$ is represented by the formula $x^{\sigma} = x\sigma \vee \overline{x}\overline{\sigma}$.

Theorem 1. On the disjunctive expansion of a Boolean function in k variables.

Any Boolean function $f(x_1,...,x_n)$ can be represented in the following form:

$$f(x_1,...,x_k,x_{k+1},...,x_n) = \bigvee_{(\sigma_1,\sigma_2,...,\sigma_k)} x_1^{\sigma_1} \wedge x_2^{\sigma_2} ... \wedge x_k^{\sigma_k} \wedge f(\sigma_1,\sigma_2,...,\sigma_k,x_{k+1},...,x_n).$$

The notation $\bigvee\limits_{(\sigma_1,\sigma_2,...,\sigma_k)}$ means multiple disjunction, which is taken over all possible sets of values $(\sigma_1,\sigma_2,...,\sigma_k)$ for any k $(1 \le k \le n)$.

Example 1. Write the disjunctive expansion of functions in variables x, y.

$$f(x,y,z,t)=\left(\overline{x\wedge y\vee \overline{z}}\right)\wedge t.$$

By the decomposition theorem we have

$$f(x,y,z,t)=\bigvee_{(\sigma_1,\sigma_2)} x_1^{\sigma_1}\wedge x_2^{\sigma_2}\wedge f(\sigma_1,y,\sigma_2,t)=\overline{x}\wedge\overline{z}\wedge f(0,y,0,t)\bigvee$$

$$\bigvee\overline{x}\wedge z\wedge f(0,y,1,t)\bigvee x\wedge\overline{z}\wedge f(1,y,0,t)\bigvee x\wedge z\wedge f(1,y,1,t).$$

In example the icon "$\wedge$" of conjunction **can not be written**, *as for multiplication:* $x\wedge z\wedge\overline{y}\wedge t=xz\overline{y}t$.

Calculate the values of the function on each set of variables (true table).

$$f(0,y,0,t)=\left(\overline{0\wedge y\vee\overline{0}}\right)\wedge t=0;$$

$$f(0,y,1,t)=\left(\overline{0\wedge y\vee\overline{1}}\right)\wedge t=t;$$

$$f(1,y,0,t)=\left(\overline{1\wedge y\vee\overline{0}}\right)\wedge t=0;$$

$$f(1,y,1,t)=\left(\overline{1\wedge y\vee\overline{1}}\right)\wedge t=\overline{y}\wedge t;$$

We substitute the obtained values in the decompositions

$$f(x,y,z,t)=\overline{x}\wedge\overline{z}\wedge 0\bigvee\overline{x}\wedge z\wedge t\bigvee x\wedge\overline{z}\wedge 0\bigvee x\wedge z\wedge\overline{y}\wedge t=$$

$$=\overline{x}\wedge z\wedge t\bigvee x\wedge z\wedge\overline{y}\wedge t.$$

<u>**Corollary 1**</u>

The disjunctive expansion of a function in one variable has the form

$$f(x_1,...,x_i,...,x_n)=\overline{x_i}\wedge f(x_1,x_2,...,0,x_{i+1},...,x_n)\bigvee x_i\wedge f(x_1,x_2,...,1,x_{i+1},...,x_n).$$

For the previous example, we have

$$f(x,y,z,t)=\overline{x}\wedge f(0,y,z,t)\bigvee x\wedge f(1,y,z,t)=\overline{x}zt\bigvee x\overline{y}zt.$$

(Check by yourself).

Corollary 2

The disjunctive expansion of the function in all n variables has the form $\left(f\left(x_1,x_2,...,x_n\right)\neq 0\right)$.

$$f\left(x_1,x_2,...,x_n\right)=\bigvee_{\substack{(\sigma_1,\sigma_2,...,\sigma_n)\\ f(\sigma_1,\sigma_2,...,\sigma_n)=1}} x_1^{\sigma_1}\wedge x_2^{\sigma_2}\wedge...\wedge x_n^{\sigma_n}.$$

Record $\bigvee\limits_{\substack{(\sigma_1,\sigma_2,...,\sigma_k)\\ f(\sigma_1,\sigma_2,...,\sigma_n)=1}}$, means that the disjunction is taken over all sets of values on which $f\left(\sigma_1,\sigma_2,...,\sigma_n\right)=1$.

Example 2. Find the disjunctive expansion of the function in all variables.

$f\left(x,y,z\right)=xy\vee z.$

Define function values for each interpretation

$\underline{f\left(0,0,0\right)}=0\wedge 0\vee\bar{0}=0\vee 1=1;$

$f\left(0,0,1\right)=0\wedge 0\vee\bar{1}=0\vee 0=0;$

$\underline{f\left(0,1,0\right)}=0\wedge 1\vee\bar{0}=0\vee 1=1;$

$f\left(0,1,1\right)=0\wedge 1\vee\bar{1}=0\vee 0=0;$

$\underline{f\left(1,0,0\right)}=1\wedge 0\vee\bar{0}=0\vee 1=1;$

$f\left(1,0,1\right)=1\wedge 0\vee\bar{1}=0\vee 0=0;$

$\underline{f\left(1,1,0\right)}=1\wedge 1\vee\bar{0}=1\vee 1=1;$

$\underline{f\left(1,1,1\right)}=1\wedge 1\vee\bar{1}=1\vee 0=1.$

Then the decomposition has the form

$$f\left(x,y,z\right)=x^0y^0z^0\vee x^0y^1z^0\vee x^1y^0z^0\vee x^1y^1z^0\vee x^1y^1z^1=$$

$$=\bar{x}\bar{y}\bar{z}\vee\bar{x}y\bar{z}\vee x\bar{y}\bar{z}\vee xy\bar{z}\vee xyz.$$

An **elementary conjunction** is a conjunction of any number of Boolean variables that are taken with or without negation, in which each variable occurs no more than once.

A **disjunctive normal form** (DNF) is a formula that is depicted as a disjunction of elementary conjunctions.

An elementary conjunction is called a **constituent of the unit** (**minterm**) of a function $f\left(x_1, x_2, ..., x_n\right)$, if $f\left(\sigma_1, \sigma_2, ..., \sigma_n\right) = 1$, that is, an interpretation that turns a given elementary conjunction into a unit, turns it into a unit and a function f.

The constituent unit has the properties:

1. The constituent unit is equal to one only on the corresponding interpretation.

2. The value of the constituent of a unit is uniquely determined by the number of the corresponding interpretation.

3. The conjunction of any number of different constituent units is zero.

A **perfect disjunctive normal form** (PDNF) of a Boolean function is a formula that is depicted as a disjunction constituent of the unit of a given function.

Example 3. Find the disjunctive decomposition of functions in variables x, z: $\left(yx \vee x\bar{z}\right)\left(x \vee \bar{y}z\left(z \vee \bar{x}y\right)\right)$.

$$f\left(x, y, z\right) = x^0 z^0 f\left(0, y, 0\right) \vee x^0 z^1 f\left(0, y, 1\right) \vee x^1 z^0 f\left(1, y, 0\right) \vee x^1 z^1 f\left(1, y, 1\right) =$$

$$= x\bar{z} \cdot 1 \vee xzy.$$

$$f\left(0, y, 0\right) = \left(y0 \vee 0 \cdot 1\right)\left(0 \vee \bar{y} \cdot 0\left(0 \vee 1y\right)\right) = 0 \cdot \left(0 \vee 0\right) = 0;$$

$$f\left(0, y, 1\right) = \left(y0 \vee 0 \cdot 1\right)\left(0 \vee \bar{y} \cdot 1\left(1 \vee 1y\right)\right) = 0;$$

$$f\left(1, y, 0\right) = \left(y \cdot 1 \vee 1 \cdot 1\right)\left(1 \vee \bar{y} \cdot 0\left(0 \vee 0y\right)\right) = \left(y \vee 1\right) \cdot 1 = 1;$$

$$f\left(1, y, 1\right) = \left(y \cdot 1 \vee 1 \cdot 0\right)\left(1 \vee \bar{y} \cdot 1\left(1 \vee 0y\right)\right) = y \cdot 1 = y;$$

Example 4. Find PDNF $\left(x\left(\bar{x} \to y\right)\right) \to y$.

$$f\left(x,y\right) = x^0 y^0 f\left(0,0\right) \vee x^0 y^1 f\left(0,1\right) \vee x^1 y^0 f\left(1,0\right) \vee x^1 y^1 f\left(1,1\right) =$$

$$= \bar{x}\bar{y} \vee \bar{x}y \vee xy.$$

$$f\left(0,0\right) = 0 \to 0 = 1;$$

$$f\left(0,1\right) = 0 \to 1 = 1;$$

$$f\left(1,0\right) = \left(1 \cdot \left(0 \to 0\right)\right) \to 0 = 1 \to 0 = 0;$$

$$f\left(1,1\right) = \left(1 \cdot \left(0 \to 1\right)\right) \to 1 = 1 \to 1 = 1.$$

Example 5. Find PDNF of the function $f\left(x,y,z\right) = \left(1,1,0,1,0,0,0,0\right)$ (Values from true table.)

xyz	*f(x,y,z)*
000	1
001	1
010	0
011	1
100	0
101	0
110	0
111	0

$$f\left(x,y,z\right) = \bar{x}\bar{y}\bar{z} \vee \bar{x}\bar{y}z \vee \bar{x}yz.$$

Example 6. Find the true table of the function, it's PDNF: $x\bar{y}\bar{z} \vee \bar{x}y\bar{z} \vee x\bar{y}z$.

xyz	*f(x,y,z)*
000	0
001	0
010	1
011	0
100	1
101	1
110	0
111	0

Control questions

1. What is a disjunctive decomposition of a Boolean function?
2. What is an elementary conjunction?
3. What is DNF?
4. What is PDNF?

<u>Section 5.</u> Conjunctive decompositions of Boolean functions

The perfect disjunctive normal form of the function is the result of the disjunctive expansion of the function in all variables and contains only operations $\wedge, \vee, \neg$.

Applying the principle of duality to PDNF, we obtain an image of a function called conjunctive decomposition.

<u>Theorem 2.</u> On the conjunctive expansion of a Boolean function $f\left(x_1, x_2, ..., x_n\right)$ in k variables.

Any Boolean function $f\left(x_1, x_2, ..., x_n\right)$ can be represented in the following form:

$$f\left(x_1, x_2, ..., x_k, x_{k+1}, ..., x_n\right) = \bigwedge_{(\sigma_1, \sigma_2, ..., \sigma_k)} x_1^{\overline{\sigma_1}} \vee x_2^{\overline{\sigma_2}} \vee ... \vee x_k^{\overline{\sigma_k}} \vee f\left(\sigma_1, \sigma_2, ..., \sigma_k, x_{k+1}, ..., x_n\right).$$

The notation $\bigwedge\limits_{(\sigma_1, \sigma_2, ..., \sigma_k)}$ means multiple conjunction, which is taken over all possible sets of values $\left(\sigma_1, \sigma_2, ..., \sigma_k\right)$ for any k.

Example 1. Write the conjunctive expansion of functions in variables x, t:

$f\left(x, y, z, t\right) = \left(xy \vee \overline{z}\right)t.$

From the theorem 2:

$$f\left(x, y, z, t\right) = \left(x^{\bar{0}} \vee t^{\bar{0}} \vee f\left(0, y, z, 0\right)\right)\left(x^{\bar{0}} \vee t^{\bar{1}} \vee f\left(0, y, z, 1\right)\right) \wedge$$

$$\wedge\left(x^{\bar{1}} \vee t^{\bar{0}} \vee f\left(1, y, z, 0\right)\right)\left(x^{\bar{1}} \vee t^{\bar{1}} \vee f\left(1, y, z, 1\right)\right).$$

$$f(0,y,z,0) = (0 \wedge y \vee \overline{z}) \wedge 0 = 0;$$

$$f(0,y,z,1) = (0 \wedge y \vee \overline{z}) \wedge 1 = \overline{z};$$

$$f(1,y,z,0) = (1 \wedge y \vee \overline{z}) \wedge 0 = 0;$$

$$f(1,y,z,1) = (1 \wedge y \vee \overline{z}) \wedge 1 = y \vee \overline{z};$$

$$f(x,y,z) = (x \vee t)(x \vee \overline{t} \vee \overline{z})(\overline{x} \vee t)(\overline{x} \vee \overline{t} \vee y \vee \overline{z}).$$

<u>Corollary 1</u>

The conjunctive expansion of a function in one variable has the form

$$f(x_1, x_2, ..., x_n) = \bigwedge_{\sigma_i} x_i^{\sigma_i} \vee f(x_1, x_2, ..., x_{i-1}, \sigma_i, x_{i+1}, ..., x_n).$$

<u>Corollary 2</u>

The conjunctive expansion of the function in all n variables has the form

$$f(x_1, x_2, ..., x_n) = \bigwedge_{\substack{(\sigma_1, \sigma_2, ..., \sigma_n) \\ f(\sigma_1, \sigma_2, ..., \sigma_n) = 0}} x_1^{\overline{\sigma_1}} \vee x_2^{\overline{\sigma_2}} \vee ... \vee x_n^{\overline{\sigma_n}}.$$

Record $\bigwedge\limits_{f(\sigma_1, \sigma_2, ..., \sigma_n) = 0}$ means that the conjunction is taken over all sets of values on which $f(x_1, x_2, ..., x_n) = 0$.

Example 2. Find the conjunctive expansion of the function in all variables.

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

Define function values for each interpretation

$$f(0,0,0) = 0 \wedge 0 \vee \overline{0} = 0 \vee 1 = 1;$$

$$f(0,0,1) = 0 \wedge 0 \vee \overline{1} = 0 \vee 0 = 0;$$

$$f(0,1,0) = 0 \wedge 1 \vee \overline{0} = 0 \vee 1 = 1;$$

$$f(0,1,1) = 0 \wedge 1 \vee \overline{1} = 0 \vee 0 = 0;$$

$$f(1,0,0)=1\wedge 0\vee \bar{0}=0\vee 1=1;$$

$$f(1,0,1)=1\wedge 0\vee \bar{1}=0\vee 0=0;$$

$$f(1,1,0)=1\wedge 1\vee \bar{0}=1\vee 1=1;$$

$$f(1,1,1)=1\wedge 1\vee \bar{1}=1\vee 0=1;$$

$$f(x,y,z)=\left(x^{\bar{0}}\vee y^{\bar{0}}\vee z^{\bar{1}}\right)\left(x^{\bar{0}}\vee y^{\bar{1}}\vee z^{\bar{1}}\right)\left(x^{\bar{1}}\vee y^{\bar{0}}\vee z^{\bar{1}}\right)=$$

$$=\left(x\vee y\vee \bar{z}\right)\left(x\vee \bar{y}\vee \bar{z}\right)\left(\bar{x}\vee y\vee \bar{z}\right).$$

An **elementary disjunction** is a disjunction of any number of Boolean variables that are taken with or without negation, in which each variable occurs no more than once.

A **conjunctive normal form** (CNF) is a formula that is depicted as a conjunction of elementary disjunctions.

An elementary disjunction $x_1^{\bar{\sigma}_1}\vee x_2^{\bar{\sigma}_2}\vee\ldots\vee x_n^{\bar{\sigma}_n}$ is called a **constituent of the zero** (**maxterm**) of a function $f(x_1,x_2,\ldots,x_n)$, if $f(\sigma_1,\sigma_2,\ldots,\sigma_n)=0$, that is, an interpretation that turns a given elementary disjunction into a zero, turns it into a zero and a function f.

The constituent zero has the properties:

1. The constituent zero is equal to zero only on the corresponding interpretation.
2. The value of the constituent of a zero is uniquely determined by the number of the corresponding interpretation.
3. The disjunction of any number of different constituents of zero is unit.

A **perfect conjunctive normal form** (PCNF) of a Boolean function is a formula that is depicted as a conjunction constituent of a zero of a given function.

The following conclusions can be drawn from the analysis of PDNF and PCNF functions:

1. For each non-zero Boolean function, there is an image in the form of a perfect disjunctive normal form.

2. For each non-unit Boolean function, there is an image in the form of a perfect conjunctive normal form.

3. Two different Boolean functions cannot have the same PDNF or PCNF.

4. For each Boolean function, there is an image in the form of a formula of Boolean algebra, which contains only the operations of disjunction, conjunction and negation.

Control questions

1. What is the conjunctive expansion of a Boolean function?
2. What is elementary disjunction?
3. What is CNF?
4. What is PCNF?

<u>Section 6.</u> The transition algorithms from PDNF and PCNF

In the previous lecture, we talked about the fact that perfect disjunctive and perfect conjunctive normal forms are obtained with the corresponding decompositions of Boolean functions in all variables.

Recall the formulas for the expansion in all variables.

Disjunctive decomposition:

$$f\left(x_1,x_2,...,x_n\right)=\bigvee_{(\sigma_1,\sigma_2,...,\sigma_n)} x_1^{\sigma_1}\wedge x_2^{\sigma_2}\wedge ...\wedge x_n^{\sigma_n};$$

$$f\left(\sigma_1,\sigma_2,...,\sigma_n\right)=1.$$

Conjunctive decomposition:

$$f\left(x_1,x_2,...,x_n\right)=\bigwedge_{(\sigma_1,\sigma_2,...,\sigma_n)} x_1^{\overline{\sigma_1}}\vee x_2^{\overline{\sigma_2}}\vee ...\vee x_n^{\overline{\sigma_n}};$$

$$f\left(\sigma_1,\sigma_2,...,\sigma_n\right)=0.$$

The perfect disjunctive normal form contains only **mintherms**. These terms correspond to interpretations on which the Boolean function is equal to one. If the function argument takes the value 0, then the corresponding variable in the PDNF is written with a negation. If the original argument is equal to one, then in PDNF is a variable without negation.

The perfect conjunctive normal form is the product of the maxterms, which correspond only to those interpretations on which the function is equal to zero.

In PCDF, variables are written without negation if in the interpretation the original argument is zero, and with negation if the original argument is one.

To make the transition from the truth table of a Boolean function to its perfect forms and vice versa is quite simple. The solutions to some important equations are given further. It`s suggested to the students to fill the coloumns in the truth tables by theirselves.

Example 1. Find the PDNF and PCNF for the function $f=\left(\overline{x\vee y}\right)\left(x\rightarrow y\right)$.

1) We build a truth table

xy	f
00	1
01	0
10	0
11	0

$$f\left(0,0\right)=\left(\overline{0\vee 0}\right)\left(0\rightarrow 0\right)=1\cdot 1=1;$$

$$f\left(0,1\right)=\left(\overline{0\vee 1}\right)\left(0\rightarrow 1\right)=\bar{1}\wedge 1=0\wedge 1=0;$$

$$f\left(1,0\right)=\left(\overline{1\vee 0}\right)\left(1\rightarrow 0\right)=\bar{1}\cdot 0=0\cdot 0=0;$$

$$f\left(1,1\right)=\left(\overline{1\vee 1}\right)\left(1\rightarrow 1\right)=\bar{1}\cdot 1=0\cdot 1=0;$$

Haw many terms does it contain PDNF, PCNF?

PDNF: $f=\bar{x}\bar{y};$

PCNF: $f=\left(x\vee\bar{y}\right)\left(\bar{x}\vee y\right)\left(\bar{x}\vee\bar{y}\right).$

Control questions

TASK. Solve problems yourself!

Example 2. Build truth tables for functions given by perfect normal formulas.

1. PDNF:

xyz	*f*1	*f*2	*f*3
000	0		
001	1		
010	0		
011	0		
100	1		
101	0		
110	0		
111	1		

a) $f_1(x,y,z)=xyz\vee x\bar{y}\bar{z}\vee \bar{x}\bar{y}\bar{z};$

b) $f_2(x,y,z)=xy\bar{z}\vee \bar{x}yz\vee x\bar{y}z;$

c) $f_3(x,y,z)=\bar{x}yz\vee x\bar{y}\bar{z}\vee \bar{x}y\bar{z}.$

2. PCNF:

xyz	*f*1	*f*2	*f*3
000	1		
001	0		
010	0		
011	1		
100	1		
101	0		
110	1		
111	1		

a) $f_1(x,y,z)=(\bar{x}\vee y\vee \bar{z})(x\vee \bar{y}\vee z)(x\vee y\vee \bar{z});$

b) $f_2(x,y,z)=(\bar{x}\vee y\vee z)(x\vee \bar{y}\vee \bar{z})(\bar{x}\vee \bar{y}\vee z);$

c) $f_3(x,y,z)=(\bar{x}\vee \bar{y}\vee \bar{z})(x\vee y\vee z)(\bar{x}\vee y\vee \bar{z});$

Example 3. Find PDNF for functions

a) $f_1(x,y,z)=(0,1,1,1,0,0,1,0);$

b) $f_2(x,y,z)=(x\wedge y)\to z;$

Truth tables:

xyz	*f*1	*f*2
000	0	
001	1	
010	1	
011	1	
100	0	
101	0	
110	1	
111	0	

$f_1(x,y,z)=\bar{x}\bar{y}z\vee \bar{x}y\bar{z}\vee \bar{x}yz\vee xy\bar{z};$

$f_2(x,y,z)=\bar{x}\bar{y}\bar{z}\vee \bar{x}\bar{y}z\vee \bar{x}y\bar{z}\vee \bar{x}yz\vee x\bar{y}\bar{z}\vee x\bar{y}z\vee xyz;$

Example 4. Find PCNF for functions

a) $f_1(x,y,z)=(1,0,0,1,0,0,0,1)$;

b) $f_2(x,y,z)=x\vee\overline{y}z\vee\overline{z}$;

Truth tables:

xyz	*f1*	*f2*
000	1	
001	0	
010	0	
011	1	
100	0	
101	0	
110	0	
111	1	

$$f_2(0,0,0)=0\vee 1\cdot 0\vee 1=1;$$
$$f_2(0,0,1)=0\vee 1\cdot 1\vee 0=1;$$
$$f_2(0,1,0)=0\vee 0\cdot 0\vee 1=1;$$
$$f_2(0,1,1)=0\vee 0\cdot 1\vee 0=0;$$
$$f_2(1,0,0)=1\vee 1\cdot 0\vee 1=1;$$
$$f_2(1,0,1)=1\vee 1\cdot 1\vee 0=1;$$
$$f_2(1,1,0)=1\vee 0\cdot 0\vee 1=1;$$
$$f_2(1,1,1)=1\vee 0\cdot 1\vee 0=1;$$

$$f_1(x,y,z)=\left(x\vee y\vee\overline{z}\right)\left(x\vee\overline{y}\vee z\right)\left(\overline{x}\vee y\vee z\right)\wedge;\left(\overline{x}\vee y\vee\overline{z}\right)\left(\overline{x}\vee\overline{y}\vee z\right);$$

$$f_2(x,y,z)=\left(x\vee\overline{y}\vee\overline{z}\right);$$

Example 5. Find the perfect disjunctive normal form for a function given by a serial number $f_{201}(x,y,z)$.

1) Convert 201 to binary (start with the highest possible power of 2).

$$201=128+64+8+1=1\cdot 2^7+1\cdot 2^6+0\cdot 2^5+0\cdot 2^4+$$
$$+1\cdot 2^3+0\cdot 2^2+0\cdot 2^1+1\cdot 2^0;$$

xyz	*f*
000	1
001	1
010	0
011	0
100	1
101	0
110	0
111	1

$$f(x,y,z)=\overline{x}\,\overline{y}\,\overline{z}\vee\overline{x}\,\overline{y}z\vee x\overline{y}\,\overline{z}\vee xyz \quad \text{(PDNF).}$$

<u>Section 7.</u> Zhegalkin's algebra

An algebra $(B,\wedge,\oplus,0,1)=0$ obtained on a set $B=\{0;1\}$ together with operations $\wedge,\oplus$ (the sum modulo 2) and constants 0,1 is called the **Zhegalkin's algebra**.

The sum modulo 2 is the remainder of dividing the sum of numbers by 2. The same operation is called *XOR*. The results of this operation are in the standard truth table. It is easy to remember if the same elements add up, the result of addition 0, if different elements are summed up, the result of addition 1.

In this algebra, each element is inverse to itself. For example, to solve an equation, it’s enough to add the same elements to both sides of the equation.

$$x\oplus a=b \ \Rightarrow\ x\oplus \underbrace{a\oplus a}_{=0}=b\oplus a \ \Rightarrow\ x=b\oplus a.$$

<u>**Basic identities**</u> of Zhegalkin's algebra

1. Commutative law

$$x\wedge y=y\wedge x,\quad x\oplus y=y\oplus x.$$

2. Associative law

$$x\wedge(y\wedge z)=(x\wedge y)\wedge z,\quad x\oplus(y\oplus z)=(x\oplus y)\oplus z.$$

3. Distributive law

$$x\wedge(y\oplus z)=xy\oplus xz.$$

4. Law of Self-absorption

$$x\wedge x=x,\quad x\oplus x=0.$$

5. Operations with 0 and 1

$$x\wedge 0=0,\quad x\wedge 1=x,\quad x\oplus 0=x.$$

From the previous lecture, we know that any Boolean function can be represented using conjunction, disjunction, and negation operations.

For the Zhegalkin's algebra, it is necessary to express negation and disjunction in terms of conjunction and a sum modulo two.

$$\bar{x} = x \oplus 1;$$
$$x \vee y = xy \oplus x \oplus y.$$

Among all equivalent images, functions in the Zhegalkin's algebra distinguish a special form of formulas - the Zhegalkin's polynomial.

The Zhegalkin's polynomial is a representation of a Boolean function in the form of a finite sum modulo two pairwise distinct elementary conjunctions on a set of variables $\{x_1, x_2, ..., x_n\}$ and possibly one.

The number of variables in any elementary conjunction is its rank. The number of different elementary conjunctions is the length of the polynomial.

Example 1. Depict the logical implication function as a Zhegalkin's polynomial $f(x, y) = x \to y.$

1) Find PDNF

$$x \to y = x^0 y^0 f(0,0) \vee x^0 y^1 f(0,1) \vee x^1 y^0 f(1,0) \vee x^1 y^1 f(1,1) =$$
$$= \bar{x}\bar{y} \vee \bar{x}y \vee xy,$$
$$\left(f(1,0) = 0\right).$$

Simplify the formula using elementary transformations, we add the second term and form the brackets.

$$\bar{x}\bar{y} \vee \bar{x}y \vee \bar{x}y \vee xy = \left(\bar{x}\bar{y} \vee \bar{x}y\right) \vee \left(\bar{x}y \vee xy\right) = \bar{x}\left(\bar{y} \vee y\right) \vee y\left(\bar{x} \vee x\right) =$$
$$= \bar{x} \vee y;$$
$$\bar{x} \vee y = \left(x \oplus 1\right) \vee y = \left(x \oplus 1\right) y \oplus \left(x \oplus 1\right) \oplus y =$$
$$= xy \oplus y \oplus x \oplus 1 \oplus y = xy \oplus x \oplus 1.$$

A Boolean function is called linear if its Zhegalkin's polynomial does not contain a conjunction of variables.

Example 2. Investigate function linearity

$$f\left(x, y, z\right) = \left(x \vee y\right) \to \bar{z}.$$

We use the known or found representations of functions through a polynomial

$$\left(x \to y = \bar{x} \vee y = xy \oplus x \oplus 1;\ \ x \vee y = xy \oplus x \oplus y;\ \ \bar{x} = x \oplus 1\right);$$

$$(x \vee y) \overset{(1)}{\to} \bar{z} = (x \vee y) \overset{(2)}{\cdot} \bar{z} \oplus (x \overset{(2)}{\vee} y) \oplus 1 = (xy \oplus x \oplus y) \cdot (z \oplus 1) \oplus xy \oplus x \oplus y \oplus 1 =$$

$$= xyz \oplus xz \oplus yz \oplus \underline{\underline{xy}} \oplus \underline{x} \oplus \underset{\sim}{y} \oplus \underline{\underline{xy}} \oplus \underline{x} \oplus \underset{\sim}{y} \oplus 1 = xyz \oplus xz \oplus yz \oplus 1.$$

Function is nonlinear.

There are **two convenient ways to construct Zhegalkin's polynomials**.

<u>**The first way:**</u> using the formula

$$f(x_1, x_2, ..., x_n) = \sum_{(\sigma_1, \sigma_2, ..., \sigma_n)} \left(x_1 \oplus \overline{\sigma_1}\right)\left(x_2 \oplus \overline{\sigma_2}\right)...\left(x_n \oplus \overline{\sigma_n}\right);$$

$$f(\sigma_1, \sigma_2, ..., \sigma_n) = 1;$$

$$\left(f(x, y, z) = \sum_{(\sigma_1, \sigma_2, \sigma_3)} \left(x_1 \oplus \overline{\sigma_1}\right)\left(x_2 \oplus \overline{\sigma_2}\right)\left(x_3 \oplus \overline{\sigma_3}\right) \right).$$

<u>Expand</u> the brackets and simplify the expression.

<u>**The second way**:</u> the method of uncertain coefficients.

We will search for the polynomial of the function $f(x, y, z)$ in the form

$$f(x, y, z) = a_0 + a_1 x + a_2 y + a_3 z + a_4 xy + a_5 xz + a_6 yz + a_7 xyz.$$

Using the truth table, we substitute sets of variable values and function values into the last equality and subsequently find the indefinite coefficients.

Example 3. It is required to find the Zhegalkin's polynomial in two ways.

$f(x, y, z) = (00010101)$

xyz	f
000	0
001	0
010	0

011	1
100	0
101	1
110	0
111	1

1) $$f(x,y,z)=(x\oplus 1)yz\oplus(x\oplus 0)(y\oplus 1)(z\oplus 0)\oplus$$
$$\oplus(x\oplus 0)(y\oplus 0)(z\oplus 0)=(x\oplus 1)yz\oplus xz(y\oplus 1)\oplus xyz=$$
$$=xyz\oplus yz\oplus xyz\oplus xz\oplus xyz=xz\oplus yz\oplus xyz;$$

2) $$f(x,y,z)=a_0+a_1x+a_2y+a_3z+a_4xy+a_5xz+a_6yz+a_7xyz;$$
$$f_2(0,0,0)=a_0=0 \;\Rightarrow\; \underline{a_0=0};$$
$$f_2(0,0,1)=a_0+a_3=0 \;\Rightarrow\; 0{+}a_3=0 \;\Rightarrow\; \underline{a_3=0};$$
$$f_2(0,1,0)=a_0+a_2=0 \;\Rightarrow\; 0{+}a_2=0 \;\Rightarrow\; \underline{a_2=0};$$
$$f_2(0,1,1)=a_0+a_2+a_3+a_6=1 \;\Rightarrow\; 0{+}0{+}0{+}a_6=1 \;\Rightarrow\; \underline{a_6=1};$$
$$f_2(1,0,0)=a_0+a_1=0 \;\Rightarrow\; 0{+}a_1=0 \;\Rightarrow\; \underline{a_1=0};$$
$$f_2(1,0,1)=a_0+a_1+a_3+a_5=1 \;\Rightarrow\; 0{+}0{+}0{+}a_5=1 \;\Rightarrow\; \underline{a_5=1};$$
$$f_2(1,1,0)=a_0+a_1+a_2+a_4=0 \;\Rightarrow\; 0{+}0{+}0{+}a_4=0 \;\Rightarrow\; \underline{a_4=0};$$
$$f_2(1,1,1)=a_0+a_1+a_2+a_3+a_4+a_5+a_6+a_7=1 \;\Rightarrow\; 1{+}1{+}a_7=1 \;\Rightarrow\; \underline{a_7=1};$$
$$f(x,y,z)=xz\oplus yz\oplus xyz.$$

Control questions

TASK. Solve problems yourself!

It is required to find the Zhegalkin's polynomial in two ways:

a) f(x,y,z)=(11101100) b) *f(x,y,z)*=(10101101) c) *f(x,y,z)*=(00101101).

<u>Section 8.</u> *Minimization of Boolean functions*

We give some more definitions that characterize the important properties of Boolean functions.

A Boolean function is called zero preserving if $f(0,0,...,0)=0$.

A Boolean function is called unit preserving if $f(1,1,...,1)=1$.

Example 1. $f(x,y,z)=x\vee\overline{y}\,\overline{z}.$

$f(1,1,...,1)=1,\; f(1,1,...,1)=1$ - unit preserving function.

A Boolean function is called monotonic if, for any sets of variable values, such that the inequality holds for the values of the function.

Example 2. $f\left(x,y\right)=x\wedge y.$

$\left(0,0\right)\le\left(0,1\right);$	$f\left(0,0\right)=0;$	$f\left(0,1\right)=0;$	$0\le 0;$
$\left(0,0\right)\le\left(1,0\right);$	$f\left(0,1\right)=0;$	$f\left(1,0\right)=0;$	$0\le 0;$
$\left(0,0\right)\le\left(1,1\right);$	$f\left(0,0\right)=0;$	$f\left(1,1\right)=1;$	$0\le 1;$
$\left(0,1\right)\le\left(1,1\right);$	$f\left(0,1\right)=0;$	$f\left(1,1\right)=1;$	$0\le 1;$
$\left(1,0\right)\le\left(1,1\right);$	$f\left(1,0\right)=0;$	$f\left(1,1\right)=1;$	$0\le 1;$

This is monotonic function.

Theorem 1

A Boolean function other than zero and unity is monotonous only if it can be represented as a formula without negations.

Example 3. $\left(\bar{x}\vee\bar{y}\right)\to\left(z\vee t\right)=\left(\overline{\bar{x}\vee\bar{y}}\right)\vee\left(z\vee t\right)=xy\vee z\vee t$.

Five sets of Boolean functions are called Post classes.

T_0 - class of functions that preserve zero;

T_1 - class of functions that preserve the unit;

S - class of self-dual functions;

M - class of monotone functions;

L - class of linear functions.

Criteria for the completeness of Post: a system of Boolean functions is complete if it contains at least one function that does not preserve zero, at least one function that does not preserve unity, at least one non-self-dual function, at least one nonmonotonic and at least one nonlinear function.

The minimization task is to find the simplest formula in accordance with the selected minimization criterion (e.g. number of variables, number of operation characters).

The problem of minimizing the number of symbols and operations on the sets of DNF and CNF is called the canonical problem of minimization. As a result, we obtain the minimum DNF and CNF.

Definition

An implicant of some function φ is a function μ such that on all interpretations, where φ = 1 and function μ = 1.

Minterms of function, elementary conjunctions included in DNF are implicants of a function.

Definition

A set C is called a complete system of implicant functions if each unit value of a function corresponds to at least one unit from the set of implicants C.

Definition

A simple implicant function is called an elementary conjunction such that no eigenpart of it is an implicant of a given function.

Definition

The disjunction of all simple implicant functions is called the abbreviated disjunctive normal form.

Definition

A finite DNF is called a DNF Boolean function, which consists only of simple implicants (not necessarily all).

The function has one abbreviated DNF and several finite.

Definition

The minimum DNF is called one of the finite forms, which corresponds to the lowest value of the minimization criterion.

The following operations are used to obtain many simple implicants.

1. The operation of incomplete disjunctive bonding

$Ax \vee A\bar{x} = A \vee Ax \vee A\bar{x}$.

2. Disjunctive absorption operation

$A \vee Ax = A$.

3. Complete disjunctive bonding

$Ax \vee A\bar{x} = A$.

Example 4. Let the function be given in the form of a perfect disjunctive normal formula $f(x,y,z) = xyz \vee \bar{x}yz \vee \bar{x}\bar{y}z \vee \bar{x}\bar{y}\bar{z}$.

We complete the transformation, as a result of which we get two different finite (dead end) forms. One of them is minimal, as it contains the least number of variables and operations.

1) $f(x,y,z) = \left(xyz \vee \bar{x}yz\right) \vee \left(\bar{x}yz \vee \bar{x}\bar{y}z\right) \vee \left(\bar{x}\bar{y}z \vee \bar{x}\bar{y}\bar{z}\right) =$

$= yz\left(x \vee \bar{x}\right) \vee \bar{x}z\left(y \vee \bar{y}\right) \vee \bar{x}\bar{y}\left(z \vee \bar{z}\right) = yz \vee \bar{x}z \vee \bar{x}\bar{y}$

2) $f(x,y,z) = \left(xyz \vee \bar{x}yz\right) \vee \left(\bar{x}\bar{y}z \vee \bar{x}\bar{y}\bar{z}\right) = yz \vee \bar{x}\bar{y}$.

For the analysis of images of Boolean functions defined through CNF according to the duality principle, the concepts of implicent, abbreviated CNF, and minimal CNF are used.

The last example shows that the use of elementary transformations is not always convenient, especially if the function is represented by a more complex formula.

Next, we consider the graphical method of minimizing functions.

This is the **method of Carnot cards and Veitch diagrams**.

In Carnot cards, the values of the variables are located in the row and column headings. Each minterm corresponds to one cell of the table. Zero or one means the value of the function in this interpretation. The values of the variables are arranged so that adjacent rows or columns of the table differ in the value of only one variable.

Interpretation order for Carnot cards $(0,0)$, $(0,1)$, $(1,1)$, $(1,0)$.

$y \backslash x$	0	1
0	$\bar{x}\bar{y}$	$x\bar{y}$
1	$\bar{x}y$	xy

$z \backslash xy$	00	01	11	10
0	$\bar{x}\bar{y}\bar{z}$	$\bar{x}y\bar{z}$	$xy\bar{z}$	$x\bar{y}\bar{z}$
1	$\bar{x}\bar{y}z$	$\bar{x}yz$	xyz	$x\bar{y}z$

Fig.1. Carnot card structure for functions of two and three variables

In the future, the tables will be filled with ones and zeros. If the minterm and function are equal to unity, then one is written into the cell, in other cases zero.

Example 5. Build a Carnot card for function

$$f(x,y,z)=\bar{x}\bar{y}\bar{z}\vee\bar{x}yz\vee\bar{x}\bar{y}z\vee xyz.$$

$z \backslash xy$	00	01	11	10
0	1	0	0	0
1	1	1	1	0

Fig.2. Carnot card for $f(x,y,z)$

After building the Carnot card, the bonding (combining) operation is performed.

The bonding operation can only be carried out with neighboring cells. In the function table of two variables there are two neighboring cells, for the function of three variables - three cells and so on.

	▨		
▨	*A*	▨	

			▨
▨		▨	*B*

C	▨		▨
▨			

Fig.3. Neighboring cells for cells A and B

The fig.3 shows the options for neighboring cells for the function of three variables.

The rule of bonding (combining) cells and recording minimal DNF.

1. Build a Carnot card.

2. Combine cells into groups (only adjacent cells with units). The group may include 2^n cells (2,4,8 ...). The group has a rectangular or square shape.

3. The bonding task is to find a set of maximum groups. The number of groups in the set should be minimal. (The more units included in the group, the better. There should be as few groups as possible).

4. Each Carnot card unit must be in at least one group. Units can be included in several groups to increase the size of these groups.

5. Each group corresponds to that implicant, the real variables of which have the same values for all cells of the group.

6. The disjunction obtained after bonding simple implicants is the minimum DNF (MDNF).

A simple implicant or minterm of a function consists of all the variables on which the function depends. If there is a function of three variables, then each minterm is from a variable, etc. If the unit is a separate group, then the minterm does not change and the minimization problem is not solved.

If the units can be combined into groups, then the number of variables in the minterm is reduced. A group of two units excludes one variable from minterm. A group of four allows you to exclude two variables and so on. Therefore, the aim is to combine units into large groups, if this is possible.

Example 6. Find MDNF for function $f\left(x,y,z\right)=\overline{x}\overline{y}\overline{z}\vee\overline{x}yz\vee\overline{x}\overline{y}z\vee xyz.$

$z \backslash xy$	00	01	11	10
0	1	0	0	0
1	1	1	1	0

A B

$A:\ \overline{x}\overline{y},\quad B: yz;$

$MDNF: f\left(x,y,z\right)=\overline{x}\overline{y}\vee yz.$

Example 7. Find MDNF for function $f\left(x,y,z\right)=\overline{x}y\overline{z}\vee xy\overline{z}\vee\overline{x}\overline{y}z\vee xyz.$

$z \backslash xy$	00	01	11	10
0	0	1	1	0
1	1	0	1	0

A B C

$f\left(x,y,z\right)=A\vee B\vee C=\overline{x}\overline{y}z\vee y\overline{z}\vee xy.$

Example 8. Find MDNF for function

$f\left(x,y,z\right)=xyz\vee xy\overline{z}\vee x\overline{y}z\vee\overline{x}\overline{y}z\vee\overline{x}y\overline{z}\vee x\overline{y}\overline{z}\vee\overline{x}\overline{y}\overline{z}.$

$z \backslash xy$	00	01	11	10
0	1	1	1	1
1	1	0	1	1

B C

A

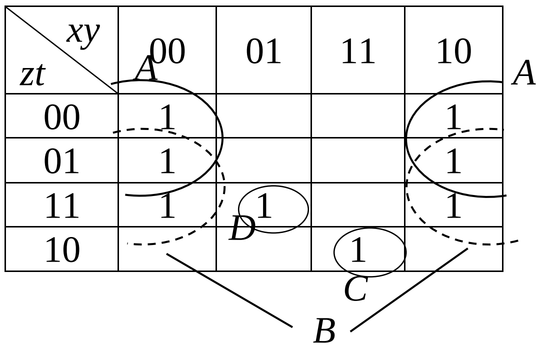
$f\left(x,y,z\right)=\overline{y}\vee\overline{z}\vee x.$

For a function of four variables, each cell has four neighboring cells. Neighboring are not only the leftmost and rightmost columns, but also the top and bottom rows.

For a function of five variables, this is a two-layer Carnot card. Each layer is from the first four variables. The first layer contains all interpretations, where the fifth variable is equal to zero, and the second layer contains the fifth variable equal to one. For each cell, five neighboring ones - four on its layer, fifth on the second layer, provided that the cells coincide when layers are applied.

Example 9. Build MDNF for function

$$f\left(x,y,z,t\right)=xyz\bar{t}\vee x\bar{y}zt\vee \bar{x}yzt\vee \bar{x}\bar{y}zt\vee x\bar{y}\bar{z}t\vee \bar{x}\bar{y}\bar{z}t\vee x\bar{y}\bar{z}\bar{t}\vee \bar{x}\bar{y}\bar{z}\bar{t}.$$



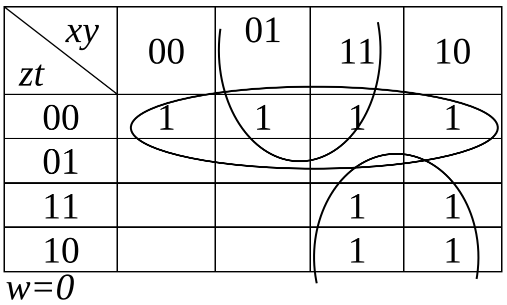
xy / zt	00	01	11	10
00	1			1
01	1			1
11	1	1		1
10			1	

$$f\left(x,y,z,t\right)=\bar{y}\bar{z}\vee \bar{y}t\vee \bar{x}zt\vee xyz\bar{t}.$$

Example 10. Build MDNF for function

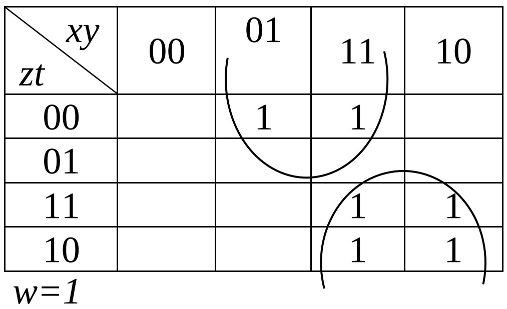
$$f\left(x,y,z,t,w\right)=\bar{x}\bar{y}\bar{z}\bar{t}\bar{w}\vee \bar{x}y\bar{z}\bar{t}\bar{w}\vee xyztw\vee xy\bar{z}\bar{t}w\vee \bar{x}y\bar{z}\bar{t}w\vee xyzt\bar{w}\vee xyz\bar{t}\bar{w}\vee$$

$$\vee x\bar{y}z\bar{t}\bar{w}\vee x\bar{y}\bar{z}\bar{t}\bar{w}\vee xy\bar{z}\bar{t}\bar{w}\vee x\bar{y}zt\bar{w}\vee xyz\bar{t}w\vee x\bar{y}z\bar{t}w\vee x\bar{y}ztw.$$



xy / zt	00	01	11	10
00	1	1	1	1
01				
11			1	1
10			1	1

w=0



xy / zt	00	01	11	10
00		1	1	
01				
11			1	1
10			1	1

w=1

$$f\left(x,y,z,t,w\right)=\bar{z}\bar{t}\bar{w}\vee y\bar{z}\bar{t}\vee xz.$$

To minimize on the set of conjunctive normal forms, Weich diagrams are used. Cells with interpretations equal to zero are marked on the cards. After gluing the cells form a minimal conjunctive normal form. The result of minimization is a conjunction of elementary disjunctions.

Example 11. Build MDNF for function

$$f\left(x,y,z,t,w\right)=\left(x\vee y\vee z\vee t\vee w\right)\cdot\left(x\vee y\vee z\vee t\vee \bar{w}\right)\cdot\left(x\vee y\vee \bar{z}\vee t\vee w\right)\cdot$$

$$\cdot\left(x\vee y\vee \bar{z}\vee \bar{t}\vee w\right)\cdot\left(x\vee y\vee \bar{z}\vee t\vee w\right)\cdot\left(x\vee \bar{y}\vee \bar{z}\vee \bar{t}\vee w\right)\cdot$$

$$\cdot\left(\bar{x}\vee y\vee z\vee t\vee w\right)\cdot\left(\bar{x}\vee y\vee z\vee t\vee \bar{w}\right)\cdot\left(\bar{x}\vee \bar{y}\vee \bar{z}\vee t\vee w\right)\cdot$$
$$\cdot\left(\bar{x}\vee y\vee \bar{z}\vee t\vee \bar{w}\right).$$

$zt \backslash xy$	00	01	11	10
00	0			0
01				
11	0	0		
10	0	0	0	

A A

w=0 B C

$zt \backslash xy$	00	01	11	10
00	0			0
01				
11				
10				0

A A D D

w=1

$$f\left(x,y,z,t,w\right)=A\wedge B\wedge C\wedge D=\left(y\vee z\vee t\right)\cdot\left(x\vee \bar{z}\vee w\right)\cdot\left(\bar{y}\vee \bar{z}\vee t\vee w\right)\cdot$$
$$\cdot\left(\bar{x}\vee y\vee t\vee \bar{w}\right).$$

If the function is ***partially defined***, it is redefined so as to obtain the most economical minimal form.

Example 12. Function $f\left(x,y,z,t\right)=1$ on sets $\left(0,0,1,0\right)$, $\left(0,1,1,0\right)$, $\left(1,0,1,0\right)$, $\left(1,0,0,0\right)$ and not defined if $xy=1$. Find MDNF.

$$f\left(x,y,z,t\right)=z\bar{t}\vee x\bar{t}$$

Example 13. Function $f\left(x,y,z,t\right)=1$ on sets $\left(0,0,1,0\right)$, $\left(0,1,1,0\right)$, $\left(1,0,1,0\right)$, $\left(1,0,0,0\right)$ and not defined if $xy=1$. Find MDNF.

$$f\left(x,y,z,t\right)=z\bar{t}\vee x\bar{t}$$

$zt \backslash xy$	00	01	11	10
00			-	1
01			-	
11			-	
10	1	1	-	1

B

A

Carnot card for a partially defined function

Control questions

1. What is the minimization criterion?

2. What are the main goals of minimization?

3. What is the meaning of minimization by the method of Carnot maps?

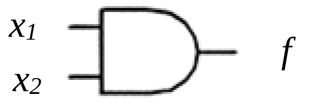
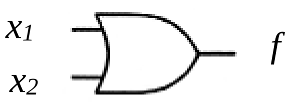
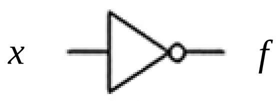
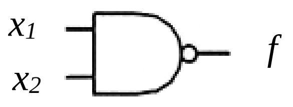
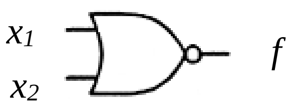
4. What is the minimization algorithm according to the Carnot method?

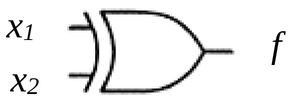
5. What is the difference between the Veitch diagram method and the Carnot method?

<u>Section 9.</u> Logic circuits

Logic circuits in computers and other electronic devices operate with sets of input and output data, which consist of zeros and ones. Boolean algebra and Boolean functions are the mathematical apparatus for working with such data and are used for the analysis and synthesis of logic circuits. The basis for constructing logic circuits is a set of logical elements. Each logic element implements some Boolean function. Its inputs correspond to Boolean variables, and the output corresponds to the value of the function. The most commonly used logical element symbols are shown in the table.

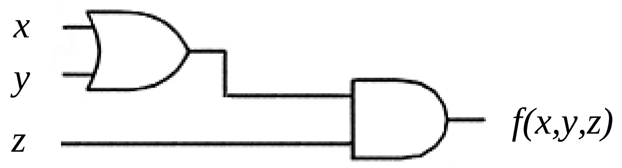
Table 1. Basic logical elements

Boolean function	Logical element symbol
$f = x_1 \wedge x_2$	
$f = x_1 \vee x_2$	
$f = \overline{x}$	
$f = \overline{x_1 \wedge x_2}$ (Schaeffer's stroke)	
$f = \overline{x_1 \vee x_2}$ (Pierce's arrow)	

$f = x_1 \oplus x_2$ (XOR)	

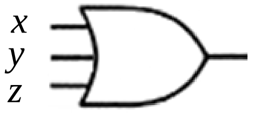
A set of logical elements is complete if it can be used to implement any Boolean function.

Example 1. Build a logical circuit that implements a function



$$f(x,y,z) = (x \vee y)z.$$

Logical elements that depict operations with the properties of associativity and commutativity on the diagrams can be depicted with the number of inputs more than two. This means repeated use of this operation. For example, the disjunction of the three elements.

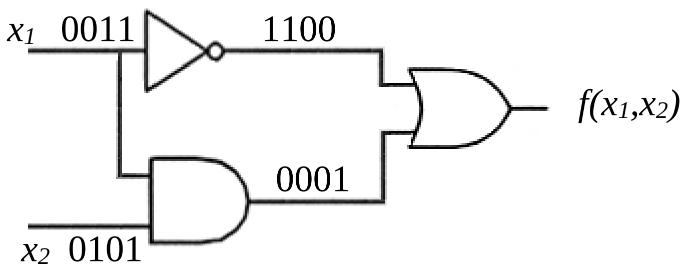


$$y = x_1 \vee x_2 \vee x_3$$

During the study of logical circuits, two main tasks arise: analysis and synthesis. The analysis of the logical circuit consists in constructing a Boolean function that this device implements. To do this, the output signal is determined on all data sets and a table of the truth of the function is compiled. Then the perfect disjunctive or conjunctive forms are built. On the other hand, by the appearance of the logical chain, you can first build the formula of the Boolean function, and then the truth table.

The synthesis task consists in constructing a logical chain for a function that is given by a truth table or formula. The value of a logical chain is its simplicity. Therefore, before constructing a logical circuit, one should obtain a minimal form of a Boolean function.

Example 2. Write a Boolean function in the form of a logical circuit. Build a minimal logic circuit.



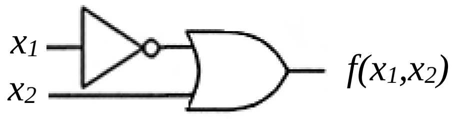
Build a true table of the function and it`s formula

$$f\left(x_1,x_2\right)=x_1x_2\vee\overline{x_1}$$

x_1	x_2	f
0	0	1
0	1	1
1	0	0
1	1	1

We minimize the formula $f\left(x_1,x_2\right)=\overline{x_1}\vee x_2$.

$x_2 \backslash x_1$	0	1
0	1	0
1	1	1



One element abbreviated.

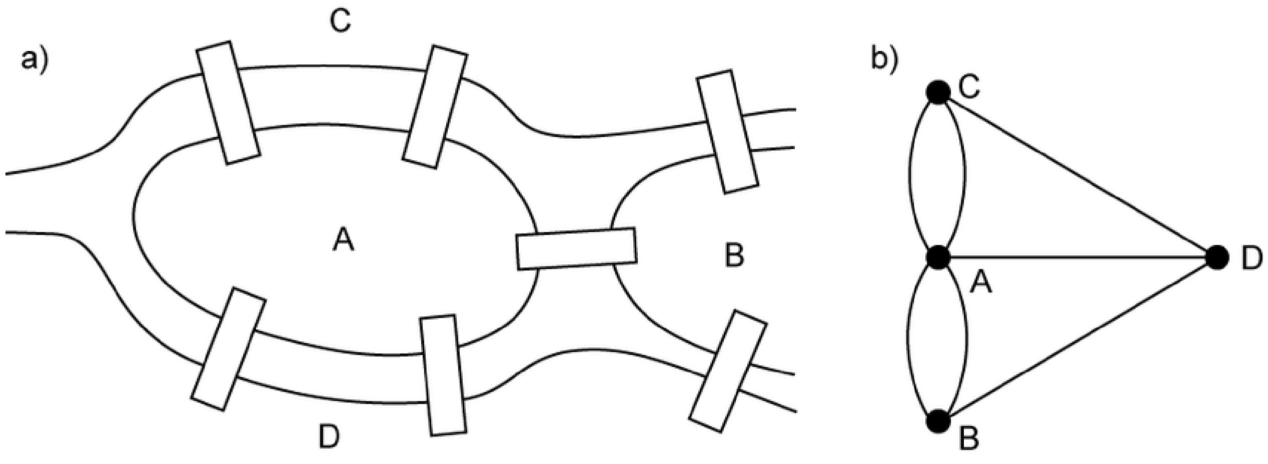
Control questions

1. In what tasks is it convenient to use logical schemes?

2. How is the transition from the Boolean algebra formula to the logical scheme performed?

<u>Section 10.</u> Graph Theory. Key Definitions

It is customary to connect the emergence of graph theory as a mathematical discipline with the work of L. Euler, in which he solves the problem of Keniksburg bridges.



Is it possible to take a walk, going through each one time and return to the starting point? With this arrangement, there are no positive solutions.

In the 19th century, Kirchhoff studied electrical circuits and separately studied the topological structure, which is now called the circuit graph. The traveling salesman task, in which it is required to visit all points, does not always have a solution. The task of the four colors is to colorize the geographical map so that any two neighboring countries are painted in different colors. (Proven using computer only).

A graph (oriented graph) is a pair of sets. Elements of the first set are points called vertices. The elements of the second set are disordered (ordered) pairs of vertices called edges (arcs).

The **set X together with the binary relation U** on this set is called the graph $G=(X,U)$.

If the points x_i, x_j are connected by an arc, then they are called **adjacent.**

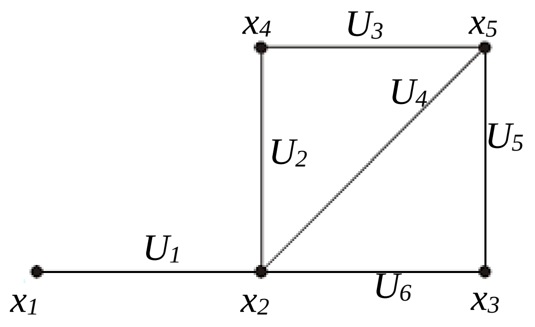
If $i=j$ then the arc (x_i, x_j) is called a loop. An edge (x_i, x_j) is called **incidental** vertices x_i and x_j. The number of edges incidental to some vertex is called the degree of the vertex. (The loop is counted twice). A vertex of degree 0 is called isolated, of degree 1 - hanging. A graph that does not have parallel (multiple) edges and loops is called simple, otherwise a multigraph. The oriented graph indicates the direction.

A graph consisting only of isolated vertices is called a null graph or an empty graph. For a directed graph, indicate the number of arcs at this vertex and the number of arcs at a vertex (half degree of entry and exit).

A sequence of vertices of a graph is called **a route (chain**) if any two neighboring vertices in this sequence are adjacent. The length of the route is equal to the number of edges that make up it.

A **simple** chain is a route that contains all the different edges. If all the vertices in the route are different, it is called an **elementary** chain. A route whose first and last vertices coincide is called a **cycle.** Like chains, there are simple and elementary cycles.

Consider the following routes.



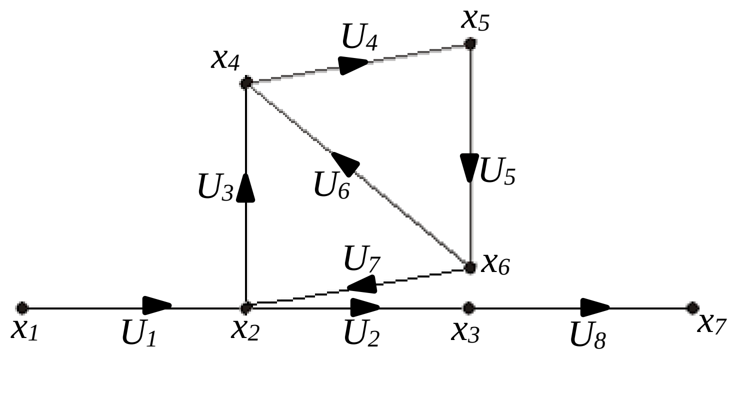
Non-simple non-elementary chain $\mu_1 = U_1U_4U_5U_6U_4U_3U_2$.

Simple elementary chain $\mu_1 = U_1U_2U_3U_5$.

Simple cycle $\mu_1 = U_2U_3U_4$.

Length of these routes $l(\mu_1)=7,\ l(\mu_2)=4,\ l(\mu_3)=3$.

For directed graph



Non-simple non-elementary path $\mu_1 = U_1U_3U_4U_5U_6U_4U_5U_7U_2U_8$.

Simple way $\mu_1 = U_1U_3U_4U_5U_7U_2U_8$.

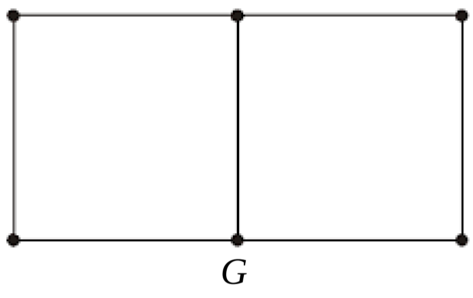
Simple elementary cycle $\mu_1 = U_3U_4U_5U_7$.

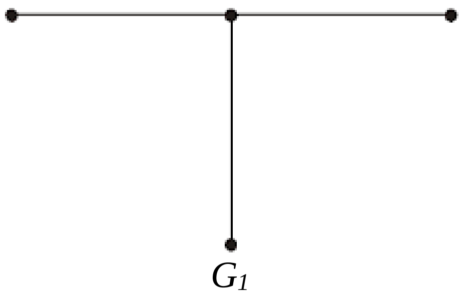
Length of these routes $l(\mu_1)=10$, $l(\mu_2)=7$, $l(\mu_3)=4$.

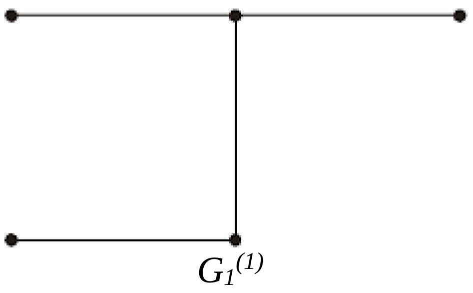
A subgraph $G_1 = (X_1, U_1)$ of a graph $G = (X, U)$ is a graph whose sets of vertices and arcs are subsets of the sets of vertices and arcs of the original graph

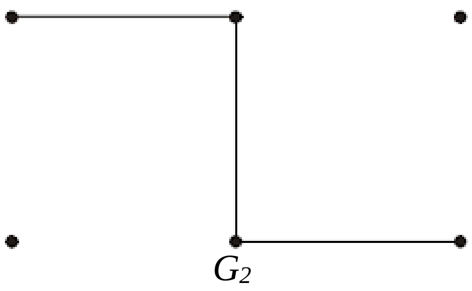
$$X_1 \subseteq X \; ; \qquad\qquad U_1 \subseteq U \, .$$

A partial graph $G_2 = (X_2, U_2)$ of a graph $G = (X, U)$ is a graph whose vertices coincide with the vertices of the graph, but some arcs are excluded.

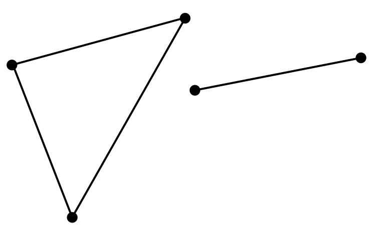

G

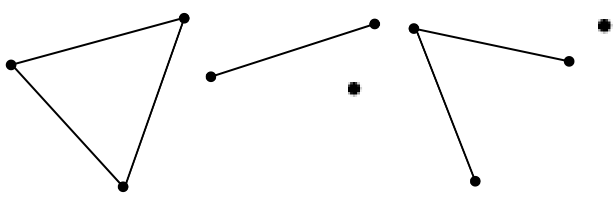

G₁


$G_1^{(1)}$

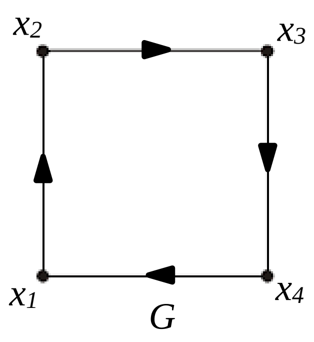

G_2

A graph is called **connected** if there is a path connecting any two of its vertices. The maximum subgraph of this graph in terms of incoming components is called its connected component. A connected graph consists of a single connected component. A disconnected graph with n vertices can contain from two to n connected components.

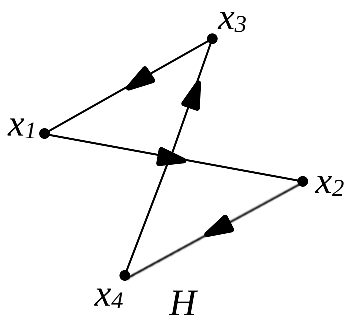

2 components


5 components

Two graphs are called **isomorphic** if they have the same number of vertices and arcs. As soon as the vertices of the first graph are connected, the corresponding pair of vertices of the second graph is connected immediately and vice versa.



$G \cong H$



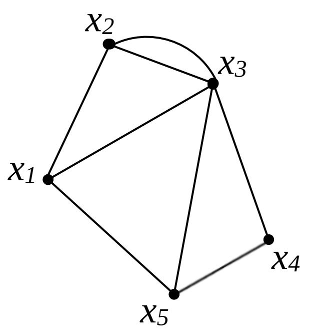
Assigning graphs methods:

1) the graph can be specified by a list of edges;

2) using an adjacency matrix or an incident matrix.

For a graph with n vertices, the adjacency matrix is a square matrix A of order n.

An element of a is equal to k if there exist k edges (arcs) that connect the vertex x_i with the vertex x_j. The element of the matrix is zero if there is no such connecting arc.



$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 & 0 \\ 1 & 2 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

<u>Matrix Properties</u>

1. For an undirected graph, the matrix is symmetric with respect to the main diagonal.

2. The sum of the elements of the row (or column) is equal to the degree of the vertex x_i.

3. If the adjacency matrix of the oriented graph is transposed (swapping rows and columns), we obtain the adjacency matrix of the graph with the opposite orientation of the arcs.

Theorem

If an adjacency matrix *A* is constructed for a directed graph, then the matrix $P = A^{\lambda}$ element p_{ij} is equal to the number of different paths of length λ from x_i to x_j.

Let a digraph have *n* vertices and *m* arcs.

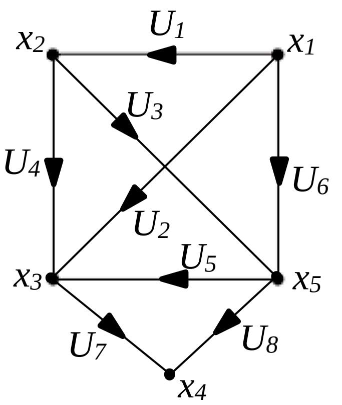
The incidence matrix of arcs of a directed graph is the matrix *B* of dimension *n*x*m,* whose elements are:

+1 if there is an arc emerging from the vertex;

-1 if the arc enters the vertex;

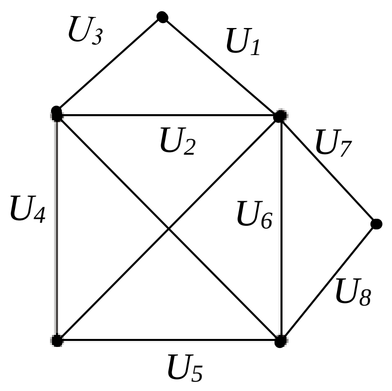
0 if the arc is not incident to the vertex.

For an undirected graph, the values in matrix *1* if the edge is incidental to the vertex or *0* if the edge is not incidental.



$$B = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & -1 & 0 & 1 & -1 & 0 & 1 \end{pmatrix}$$

The matrix of cycles is the matrix *C* in which each simple cycle corresponds to a row, and to each edge - a column. The element of the matrix is *1*, if the edge U_j enters the cycle i, 0 if the edge U_j does not enter the cycle μ_j



$\mu_1 = (U_1, U_2, U_3)$; $\mu_2 = (U_2, U_4, U_5, U_6)$;

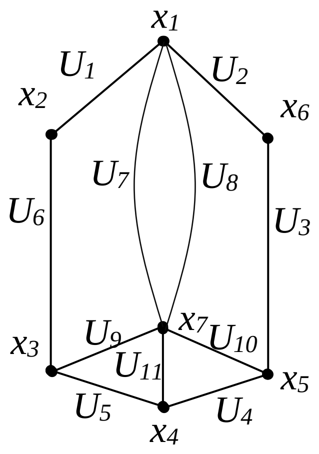
$\mu_3 = (U_6, U_7, U_8)$; $\mu_4 = (U_1, U_3, U_4, U_5, U_6)$;

$\mu_5 = (U_2, U_4, U_5, U_8, U_7)$;

$\mu_6 = (U_1, U_3, U_4, U_5, U_8, U_7)$.

$$C = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Example 1. Find the degree of the vertices of an undirected graph. Build adjacency matrix and incidence matrix.



1) $\deg x_1 = 4, \quad \deg x_2 = 2, \quad \deg x_3 = 3,$

$\deg x_4 = 3, \quad \deg x_5 = 3, \quad \deg x_6 = 2,$

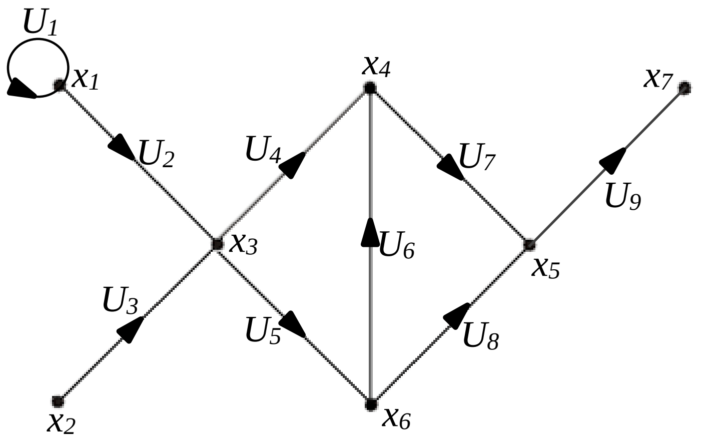
$\deg x_7 = 5;$

2) $$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 2 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix};$$

$$3)\ B = \begin{array}{c|ccccccccccc} & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 & u_9 & u_{10} & u_{11} \\ \hline x_1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ x_2 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ x_3 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ x_4 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ x_5 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ x_6 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_7 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{array}$$

Each edge connects two vertices of the graph.

Example 2. Build adjacency matrix and incidence matrix for the directed graph.



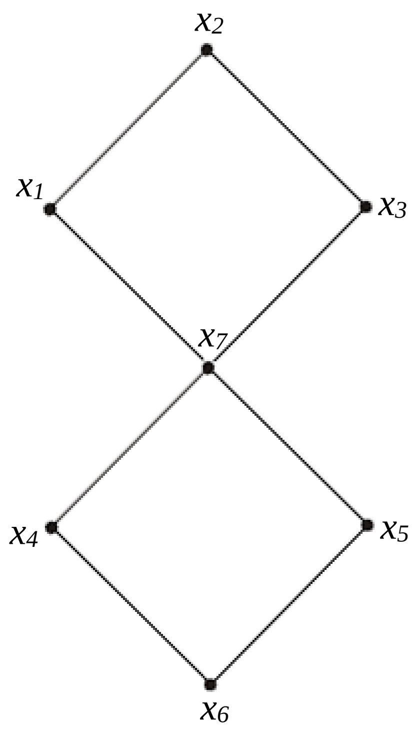
$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix};$$

$$B = \begin{pmatrix} \pm1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix}$$

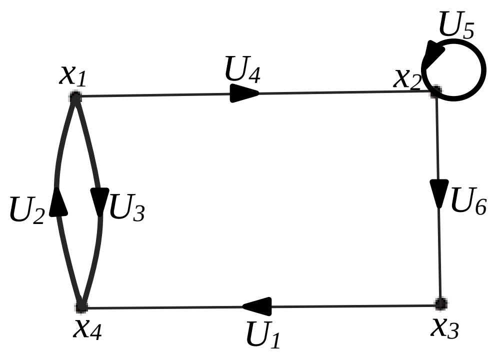
The arc leaves the top and enters the top.

Example 3. The inverse problem. By the type of matrix, restore the graph.

1) If the matrix A is symmetric, then the desired graph is undirected.
2) If the matrix B contains plus and minus one, then the graph is oriented.



$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$



$$B = \begin{pmatrix} 0 & -1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & \pm 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & -1 \\ -1 & 1 & -1 & 0 & 0 & 0 \end{pmatrix}$$

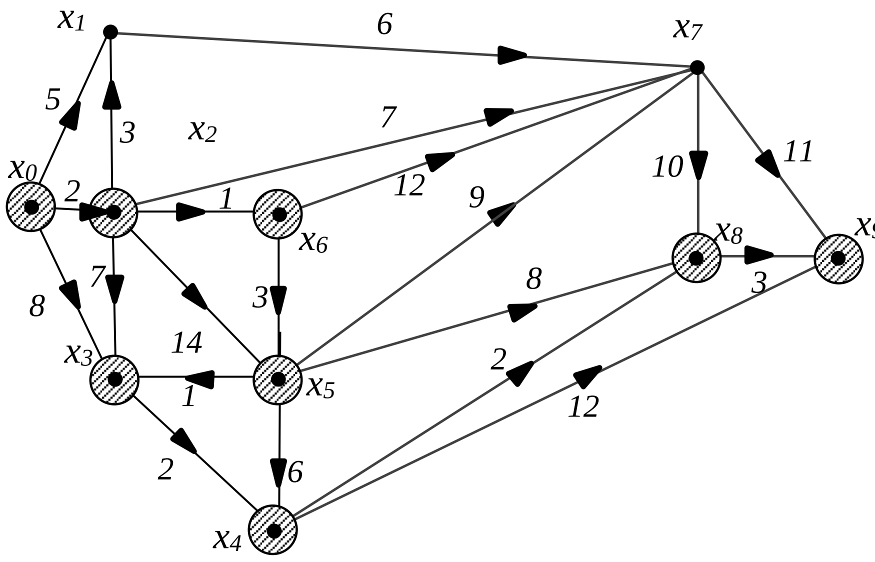
Control questions

1. What is a graph? What are the types of graphs?
2. What types of routes exist in graphs?
3. What are the ways to assign graphs?
4. What is an adjacency matrix?
5. What is an incidence matrix?

<u>Section 11.</u> *The problem of finding the shortest path in a graph*

Let each arc of the graph correspond to a certain number $l \geq 0$ called the length of the arc. These are the so-called loaded or weighted graphs. Numbers can carry various information. This can be the length of the path, the throughput of the pipeline, various time indicators, etc.

In such an oriented graph, the length of the path that connects the vertices x_0 and z is calculated by summing the lengths of all the arcs that make up the path. In this problem, you need to find the shortest path from point x_0 to point x_9.



To solve the problem, we will use one of the options of Dijkster’s algorithms.

The method consists in assigning and changing vertex labels and choosing optimal results.

1. At the first step, the starting point is the **current** one, it is assigned a label of 0, all other vertices are labeled ∞.

2. In the next steps, labels are reassigned, *the smaller the vertex label the better*.

Tags change according to the formula

$$\lambda(x_i) = \min \left\{ \lambda(x_i), \lambda(x_{current}) + l(x_{current}, x_i) \right\}.$$

The next current point is the one with the smallest label. It can be shaded.

In our diagram, we can move from the current point to points. The labels of these points vary. Other tags ∞.

$\lambda(x_1)=\min\{\lambda(x_1),\lambda(x_0)+l(x_0,x_1)\}=\min\{\infty,0+5\}=5$;

$\lambda(x_2)=\min\{\lambda(x_2),\lambda(x_0)+l(x_0,x_1)\}=\min\{\infty,0+2\}=2$;

$\lambda(x_3)=\min\{\lambda(x_3),\lambda(x_0)+l(x_0,x_3)\}=\min\{\infty,0+8\}=8$;

$\lambda(x_4)=\min\{\lambda(x_4),\lambda(x_0)+l(x_0,x_4)\}=\min\{\infty,0+\infty\}=\infty$;

....

Next current point x_2. Change labels again where possible.

$\lambda(x_1)=\min\{\lambda(x_1),\lambda(x_2)+l(x_2,x_1)\}=\min\{5,2+3\}=5$;

$\lambda(x_3)=\min\{\lambda(x_3),\lambda(x_2)+l(x_2,x_3)\}=\min\{8,2+7\}=8$;

$\lambda(x_4)=\min\{\lambda(x_4),\lambda(x_2)+l(x_2,x_4)\}=\min\{\infty,2+\infty\}=\infty$;

$\lambda(x_5)=\min\{\lambda(x_5),\lambda(x_2)+l(x_2,x_5)\}=\min\{\infty,2+14\}=16$;

$\lambda(x_6)=\min\{\lambda(x_6),\lambda(x_2)+l(x_2,x_6)\}=\min\{\infty,2+1\}=3$;

$\lambda(x_7)=\min\{\lambda(x_7),\lambda(x_2)+l(x_2,x_7)\}=\min\{\infty,2+7\}=9$;

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_2)+l(x_2,x_8)\}=\min\{\infty,\infty\}=\infty$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_2)+l(x_2,x_9)\}=\min\{\infty,\infty\}=\infty$;

$\min\{5,8,16,3,9,\infty\}=3$. We will paint x_6. $\underline{\lambda(x_6)=3}$

Next current point x_6.

$\lambda(x_1)=\min\{\lambda(x_1),\lambda(x_6)+l(x_6,x_1)\}=\min\{5,3+\infty\}=5$;

$\lambda(x_3)=\min\{\lambda(x_3),\lambda(x_6)+l(x_6,x_3)\}=\min\{8,3+\infty\}=8$;

$\lambda(x_4)=\min\{\lambda(x_4),\lambda(x_6)+l(x_6,x_4)\}=\min\{\infty,3+\infty\}=\infty$;

$\lambda(x_5)=\min\{\lambda(x_5),\lambda(x_6)+l(x_6,x_5)\}=\min\{16,3+3\}=6$;

$\lambda(x_7)=\min\{\lambda(x_7),\lambda(x_6)+l(x_6,x_7)\}=\min\{9,3+12\}=9$;

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_6)+l(x_6,x_8)\}=\min\{\infty,3+\infty\}=\infty$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_6)+l(x_6,x_9)\}=\min\{\infty,\infty\}=\infty$;

$\min\{5,8,6,9,\infty\}=5$. We will paint x_1. $\underline{\lambda(x_1)=5}$

Next current point x_1.

$\lambda(x_3)=\min\{\lambda(x_3),\lambda(x_1)+l(x_1,x_3)\}=\min\{8,5+\infty\}=8$;

$\lambda(x_4)=\min\{\lambda(x_4),\lambda(x_1)+l(x_1,x_4)\}=\min\{\infty,5+\infty\}=\infty$;

$\lambda(x_5)=\min\{\lambda(x_5),\lambda(x_1)+l(x_1,x_5)\}=\min\{6,5+\infty\}=6$;

$\lambda(x_7)=\min\{\lambda(x_7),\lambda(x_1)+l(x_1,x_7)\}=\min\{9,5+6\}=9$;

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_1)+l(x_1,x_8)\}=\min\{\infty,5+\infty\}=\infty$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_1)+l(x_1,x_9)\}=\min\{\infty,5+\infty\}=\infty$;

$\min\{\infty,\infty,6,9,\infty,\infty\}=6$. We will paint x_5. $\underline{\lambda(x_5)=6}$

Next current point x_5.

$\lambda(x_3)=\min\{\lambda(x_3),\lambda(x_5)+l(x_5,x_3)\}=\min\{8,6+1\}=7$;

$\lambda(x_4)=\min\{\lambda(x_4),\lambda(x_5)+l(x_5,x_4)\}=\min\{\infty,6+6\}=12$;

$\lambda(x_7)=\min\{\lambda(x_7),\lambda(x_5)+l(x_5,x_7)\}=\min\{9,6+9\}=9$;

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_5)+l(x_5,x_8)\}=\min\{\infty,6+8\}=14$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_5)+l(x_5,x_9)\}=\min\{\infty,6+\infty\}=\infty$;

$\min\{7,12,9,14,\infty\}=7$. We will paint x_3. $\underline{\lambda(x_3)=7}$

Next current point x_3.

$\lambda(x_4)=\min\{\lambda(x_4),\lambda(x_3)+l(x_3,x_4)\}=\min\{12,7+2\}=9$;

$\lambda(x_7)=\min\{\lambda(x_7),\lambda(x_3)+l(x_3,x_7)\}=\min\{9,\infty\}=9$;

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_3)+l(x_3,x_8)\}=\min\{14,7+\infty\}=14$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_3)+l(x_3,x_9)\}=\min\{\infty,7+\infty\}=\infty$;

$\min\{9,9,14,\infty\}=9$. We will paint x_4. $\underline{\lambda(x_4)=7}$

Next current point x_4.

$\lambda(x_7)=\min\{\lambda(x_7),\lambda(x_4)+l(x_4,x_7)\}=\min\{9,\infty\}=9$;

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_4)+l(x_4,x_8)\}=\min\{14,9+2\}=11$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_4)+l(x_4,x_9)\}=\min\{\infty,9+12\}=21$;

$\min\{9,11,21\}=9$. We will paint x_7. $\underline{\lambda(x_7)=9}$

Next current point x_7.

$\lambda(x_8)=\min\{\lambda(x_8),\lambda(x_7)+l(x_7,x_8)\}=\min\{11,9+10\}=11$;

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_7)+l(x_7,x_9)\}=\min\{21,9+11\}=20$;

$\min\{11,20\}=11$. We will paint x_8. $\underline{\lambda(x_8)=11}$

Next current point x_8.

$\lambda(x_9)=\min\{\lambda(x_9),\lambda(x_8)+l(x_8,x_9)\}=\min\{20,11+3\}=14$;

$\lambda(x_9)-\lambda(x_7)=14-9=5\neq 11;$

$\lambda(x_9)-\lambda(x_8)=14-11=3=l;$

$\lambda(x_8)-\lambda(x_7)=11-9=2\neq 10;$

$\lambda(x_8)-\lambda(x_4)=11-9=2;$

…………………………………

$\lambda(x_3)-\lambda(x_5)=1;$

$\lambda(x_5)-\lambda(x_6)=3.$

The label of the last vertex is equal to the length of the shortest path. To specify this path exactly (especially of you worked with a large graph), you must return from the endpoint to the starting one.

The return occurs along the current points (they are filled). In this case, the following condition must be fulfilled:

$$\lambda(x_N)-\lambda(x_{N-1})=l(x_{N-1},x_N).$$

The difference between the labels of these points must coincide with the length of arc between them.

Control questions

1. What is the main idea of Dijkster's algorithm?
2. What formula exists for choosing the shortest path?
3. What is the shortest path?

Section 12. ***The trees. Definitions and properties***

In graph theory, a **tree** is an undirected graph in which any two vertices are connected by *exactly one* path, or equivalently a connected acyclic undirected graph. A **forest** is an undirected graph in which any two vertices are connected by *at* *most one* path, or equivalently an acyclic undirected graph, or equivalently a disjoint union of trees.

The various kinds of data structures referred to as trees in computer science have underlying graphs that are trees in graph theory, although such data structures are generally **rooted trees**.

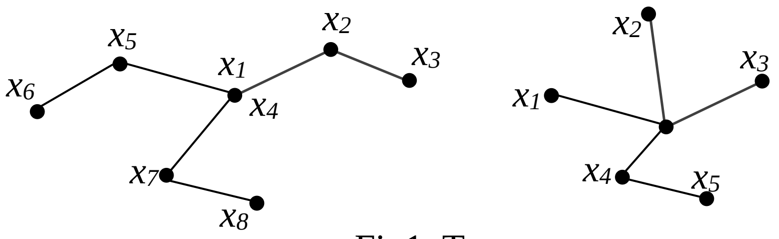


Fig1. Trees

Trees

The ***tree*** is an undirected graph G that satisfies any of the following equivalent conditions:

- G is connected and acyclic (contains no cycles).
- G is acyclic, and a simple cycle is formed if any edge is added to G.
- G is connected, but would become disconnected if any single edge is removed from G.
- G is connected and the 3-vertex complete graph K_3 is not a minor of G.
- Any two vertices in G can be connected by a unique simple path.

If G has finitely many vertices, say n of them, then the above statements are also equivalent to any of the following conditions:

- G is connected and has $n - 1$ edges.
- G is connected, and every subgraph of G includes at least one vertex with zero or one incident edges. (That is, G is connected and 1-degenerate.)
- G has no simple cycles and has $n - 1$ edges.

As elsewhere in graph theory, the order-zero graph (graph with no vertices) is generally not considered to be a tree: while it is vacuously connected as a graph (any two vertices can be connected by a path), it is not 0-connected (or even (−1)-connected) in algebraic topology, unlike non-empty trees, and violates the "one more vertex than edges" relation. It may, however, be considered as a forest consisting of zero trees.

An **internal vertex** (or **inner vertex** or **branch vertex**) is a vertex of degree at least 2. Similarly, an **external vertex** (or *outer vertex*, *terminal vertex* or *leaf*) is a vertex of degree 1.

An *irreducible tree* (or *series-reduced tree*) is a tree in which there is no vertex of degree 2.

The Forest

A ***forest*** is an undirected graph in which any two vertices are connected by at most one path. Equivalently, a forest is an undirected acyclic graph. Equivalently, a forest is an undirected graph, all of whose connected components are trees; in other words, the graph consists of a disjoint union of trees. As special cases, the order-zero graph (a forest consisting of zero trees), a single tree, and edgeless graph, are examples of forests.

The Polytree

A *polytree* (or *directed tree* or *oriented tree* or *singly connected network*) is a directed acyclic graph (DAG) whose underlying undirected graph is a tree. In other words, if we replace its directed edges with undirected edges, we obtain an undirected graph that is both connected and acyclic.

Some authors restrict the phrase "directed tree" to the case where the edges are all directed towards a particular vertex, or all directed away from a particular vertex (see arborescence).

<u>The Rooted tree</u>

A ***rooted tree*** is a tree in which one vertex has been designated the ***root***. The edges of a rooted tree can be assigned a natural orientation, either ***away from*** or ***towards*** the root, in which case the structure becomes a ***directed rooted tree***. When a directed rooted tree has an orientation away from the root, it is called an ***arborescence*** or ***out-tree***; when it has an orientation towards the root, it is called an ***anti-arborescence*** or ***in-tree***.

In a context where trees are supposed to have a root, a tree without any designated root is called a ***free tree***.

A ***labeled tree*** is a tree in which each vertex is given a unique label. The vertices of a labeled tree on *n* vertices are typically given the labels 1, 2, ..., *n*. A ***recursive tree*** is a labeled rooted tree where the vertex labels respect the tree order (i.e., if $u < v$ for two vertices *u* and *v*, then the label of *u* is smaller than the label of *v*).

In a rooted tree, the ***parent*** of a vertex *v* is the vertex connected to *v* on the path to the root; every vertex has a unique parent except the root which has no parent. A ***child*** of a vertex *v* is a vertex of which *v* is the parent.

The number of different trees that can be built on n numbered vertices is calculated by the Cayley's formula $N = n^{n-2}$.

A tree can be **encoded** in sets of zeros and ones. Consider a tree on a plane. Starting at some vertex, we move along the edges of the tree. We turn at each vertex to the edge nearest to the right and turn back at the end vertices of the tree.

We write 0 when moving along the edge for the first time and 1 when moving along the edge for the second time (in the opposite direction). If m is the number of edges of the tree, then after 2m steps we return to the starting point, passing twice along each edge. The resulting sequence of 0 and 1 (tree code) of length 2m allows you to restore not only the tree itself, but also its location on the plane. An arbitrary tree corresponds to several such codes. The frame (skeleton) of a connected graph G is any subgraph of it that contains all the vertices of G and is a tree.

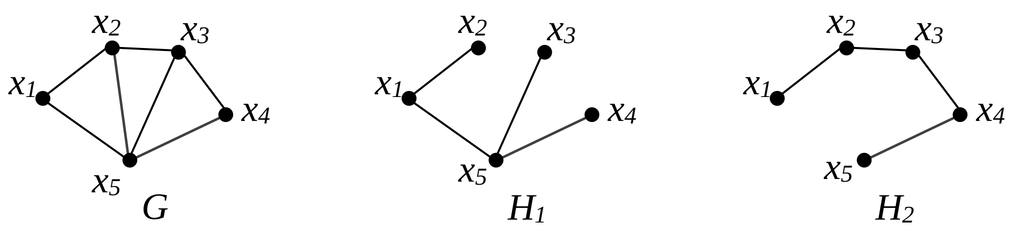


Fig 2. Sceletons

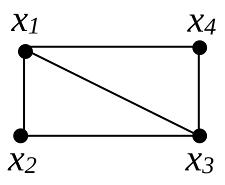
We denote by M the matrix with elements m obtained from the adjacency matrix A of the graph by replacing the signs of all elements with opposite signs. Elements of the main diagonal are replaced by the degrees of the corresponding vertices.

$$m_{ij} = \begin{cases} -a_{ij}, \text{ if } i \neq j, \ (a_{ij} - \text{ element adjacyncy matrix A}) \\ \deg x_{ii}, \text{ if } i = j. \end{cases}$$

The Matrix Tree Theorem

Let G be a connected graph without loops and multiple edges. Then all algebraic additions to the elements of the matrix M will be equal to each other and their common value gives the number of frames (skeletons) of the graph G.

Example 1. Find the number of skeletons (frames) in the graph

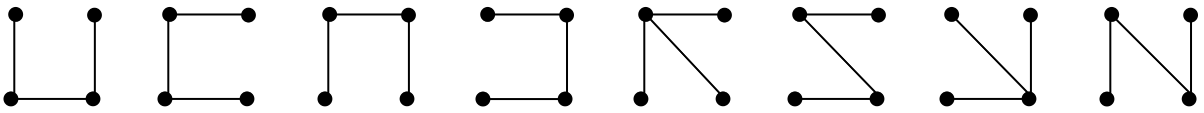


Adjacency matrix:

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \quad M = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{pmatrix}.$$

Find the algebraic complement (or addition) to the element m_{23}. This is a minor given the sign.

$$M_{23} = (-1)^{2+3}\begin{vmatrix} 3 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & 0 & 2 \end{vmatrix} = -6(-6-1+1-2) = 8.$$



The shortest tree problem

Let several cities are connected by airlines. It is required to choose a route so that from any city you can get to another (with transfers) by the shortest path.

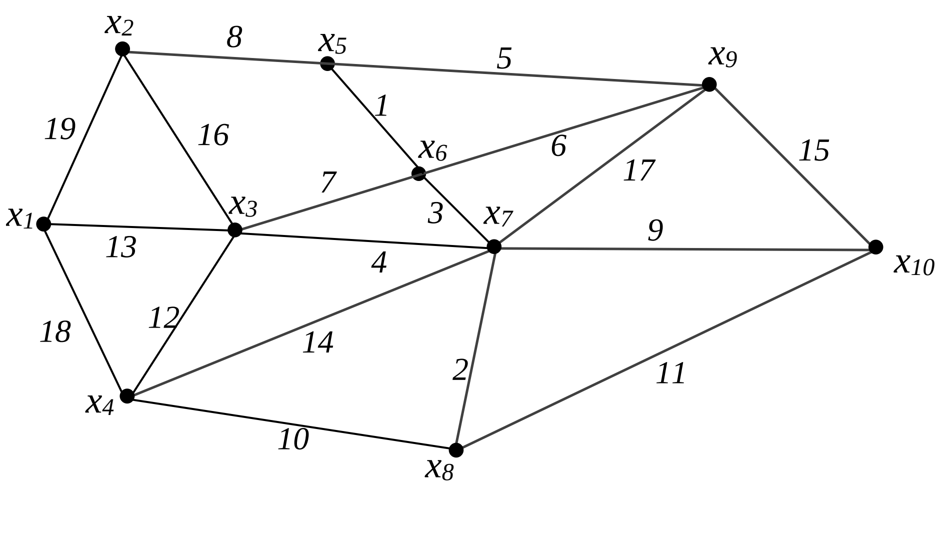
The shortest graph tree search algorithm

1. Select the shortest edge in the graph.

2. At each next stage, construct a part of the tree by adding the shortest edge from those that remained in the graph. Moreover, cycles should not form. Edges with the same minimum lengths are all included in the tree, except for those that form cycles.

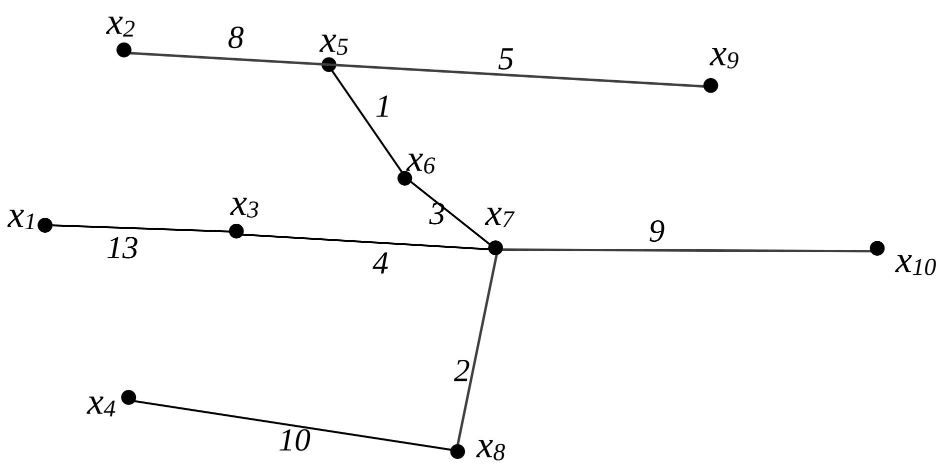
3. After the n-1 step, the process ends. (n is the number of vertices in the graph). As a result, we get a tree that does not contain cycles and has n-1 edges.

***Example 1**:*



1) (x_5, x_6) - Choose the shortest edge.
2) (x_7, x_8) - Next shortest edge.
3) (x_6, x_7) - Next shortest edge.
4) (x_3, x_7)
5) (x_5, x_9)
6) (x_6, x_9) - We don't select, as it forms cycle.
7) (x_3, x_6) - We don't select, as it forms cycle.
8) (x_2, x_5)
9) (x_7, x_{10})
10) (x_4, x_8)
11) (x_8, x_{10})

<u>**The shortest tree**</u>



It's length is 1+2+3+4+5+8+9+10+13=55.

The **Euler path** is a march in an undirected graph that passes **once** along all edges of the graph. If such a closed route it is called the Eulerian cycle. A graph in which there is an Eulerian cycle is called an Eulerian graph.

Theorem. In order for a finite connected graph to have an Euler cycle, it is necessary and sufficient that the degrees of all its vertices are even.

The **Hamiltonian path** in a graph is a route that crosses each vertex of the graph only once. If the path is closed, then this is a Hamiltonian cycle. A graph in which there is such a cycle is called a Hamiltonian graph. There is no general criterion for determining a Hamiltonian graph.

The task of building a single wireframe (skeleton) tree is one of the most important in practice. At the same time, this is the simplest task for algorithmization and computer implementation. The most famous algorithm is depth-first search (it has linear complexity cn). The algorithm goes through all the edges of the graph and all its vertices one by one. The next algorithm is Breadth First Search. At each current vertex, all incident edges and their end vertices are considered. (environment of the current vertex) For a weighted or loaded graph, the problem of finding the skeleton of the least weight is relevant. Such a task is encountered in the design of computer and cable networks, pipelines, roads, etc. If a connected graph is close to a complete graph (contains all possible edges and loops), then the number of skeleton trees is equal to n^{n-2} and direct enumeration of all possible trees and their weights has a very large amount of computations (for $n = 22$, the number of skeletons is more than 10^{25})

There are several efficient classical algorithms. For example, J. Kruskal's algorithm. The idea behind the method is that at each step, the shortest edge (with the least weight) is selected, which does not form a cycle with the previous edges.

First of all, the traveling salesman problem is associated with the concept of the **Hamiltonian graph**. The general task of a traveling salesman is as follows: using a given system of roads to visit all points or cities in such a sequence that the route traveled is the shortest. In the language of graph theory, such a problem is posed as follows: in a loaded connected graph, find the shortest route that passes through all the vertices of the graph. An additional condition for a closed route is possible. This general traveling salesman problem always has a solution.

There is one more formulation of the previous problem, in which it is additionally required that the salesman visits each point **only once**.

This is the **Hamiltonian traveling salesman problem**. She doesn't always have a solution. The same problems can be formulated for directed graphs. The computational complexity of such tasks is very high, it is on the order of (n-1)!
The exact algorithms that guarantee getting the salesman's route in any case are complex and apply to small graphs. Approximation algorithms are applied to large graphs. However, they can lead to a sub-optimal route.

Control questions

1. Which graph is called a tree?
2. What does the matrix tree theorem calculate?
3. Which graphs are called Eulerian or Hamiltonian?
4. Specify one of the methods of finding the shortest tree in a graph?

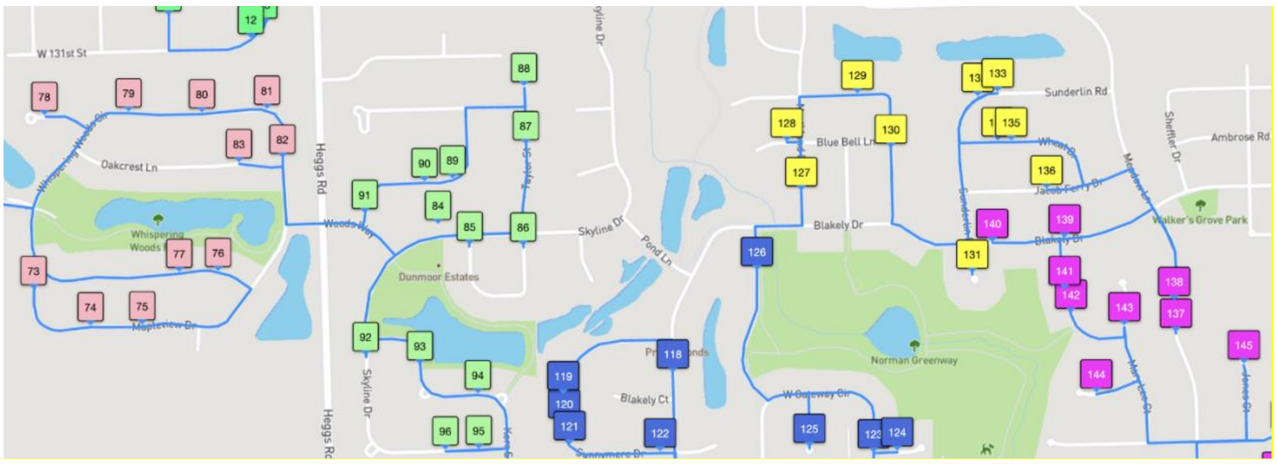
Section 13. Some applications of the graph theory

13.1. The traveling salesman problem (TSP) is an algorithmic problem tasked with finding the shortest route between a set of points and locations that must be visited. In the problem statement, the points are the cities a salesperson might visit. The salesman‘s goal is to keep both the travel costs and the distance traveled as low as possible.

Focused on optimization, TSP is often used in computer science to find the most efficient route for data to travel between various nodes. Applications include identifying network or hardware optimization methods. It was first described by Irish mathematician **W.R. Hamilton** and British mathematician **Thomas Kirkman** in the 1800s through the creation of a game that was solvable by finding a Hamilton cycle, which is a non-overlapping path between all nodes.

TSP has been studied for decades and several solutions have been theorized. The simplest solution is to try all possibilities, but this is also the most time consuming and expensive method. Many solutions use heuristics, which provides probability outcomes. However, the results are approximate and not always optimal. Other solutions include branch and bound, Monte Carlo and Las Vegas algorithms.

Rather than focus on finding the most effective route, TSP is often concerned with finding the cheapest solution. In TSPs, the large amount of variables creates a challenge when finding the shortest route, which makes approximate, fast and cheap solutions all the more attractive.



Routing a package-carrying van to your doorstep is a classic example of the traveling salesman problem. Few companies know more about this than *Amazon*. Tens of thousands of their drivers hit the road every day, each taking a Prime van in a TSP tour through 150 or more customer stops, starting and ending at a depot station. And behind the scenes an Amazon research division is constantly looking for ways to improve the efficiency and safety of their routes. The last mile in package delivery.

13.2. The transport network. The transport network, or the transportation network is a network or graph in geographic space, describing an infrastructure that permits and constrains movement or flow. Examples include but are not limited to road networks, railways, air routes, pipelines, aqueducts, and power lines.

The basic definitions

A network is

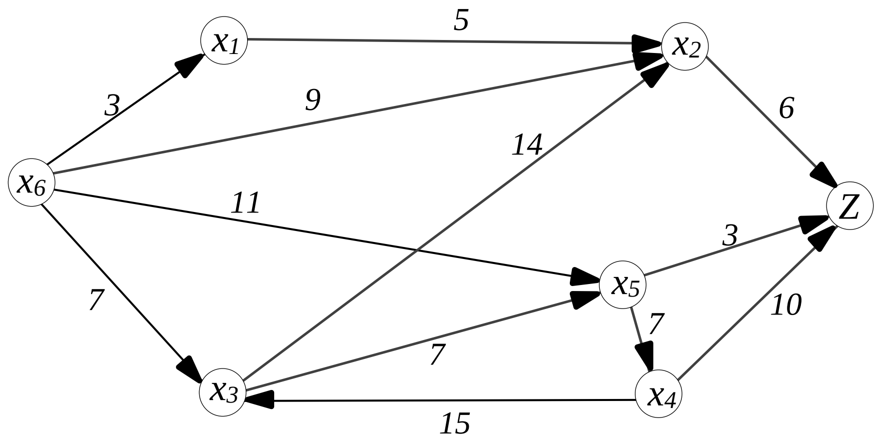
- a connected directed graph without loops.

  Denote network by $G(X,U)$, where

  X – the set of vertices,

  U – the set of arcs.

- there is only one vertex with zero number of arcs entering it. This vertex is called the **source**.
- there is only one vertex z with zero number of arcs leaving it. This vertex is called the **sink**.



Each arc $u \in U$ is supplied with the nonnegative real number $C(u)$, called **the capacity of the arc**.

The vertex $x_i \neq x_0$; $x_i \neq z$ is called **the internal vertices**.

The ***flow*** in the arc is a function $\varphi(u)$.

Properties of the flow $\varphi(u)$:

1. $0 \leq \varphi(u) \leq C(u)$;
2. $\sum\limits_{u \in U_x^-} \varphi(u) = \sum\limits_{u \in U_x^+} \varphi(u)$, $x \neq x_0$; $x \neq z$;

where U_x^- the set of arcs, entering the vertex x; U_x^+ the set of arcs, leaving the vertex x.

A flow in the arc u can be considered as the velocity in which material is transported through the arc u, so material is not accumulated in vertices of the network.

Therefore the amount of the material, that is transported from the source is equal to the sum amount of the material, that is transported into the sink.

3. $\sum_{u \in U_x^+} \varphi(u) = \sum_{u \in U_x^-} \varphi(u) = \varphi_z$,

where $\varphi(u)$ corresponds to the **value of a flow** in the network.

Let's consider the task:

A – an arbitrary set of vertices $x_0 \notin A$ and $z \in A$;

$\overline{A}$– complement of set A, $z \in A$, but $x_0 \notin \overline{A}$

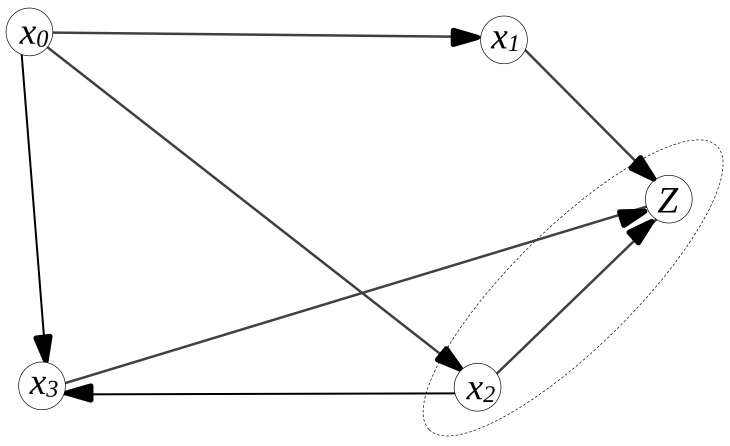
U_A^- – the set of arcs, entering to the set A;

U_A^+ – the set of arcs, leaving to the set A.

The set U_A^- is called **a cut** of a network. A cut contains arcs that terminate in the set $\overline{A}$ or ordinate in the set A.

Example:

$A = \{x_2, z\}$, $\overline{A} = \{x_0, x_1, x_3\}$ a cut $U_A^- = \{(x_1, z), (x_3, z), (x_0, x_2)\}$.



4. $\varphi_z = \sum_{u \in U_x^+} \varphi(u) - \sum_{u \in U_x^-} \varphi(u)$. The capacity of the cut is the sum of arcs capacities, belonging to this cut.

5. $\varphi_z \le C(A)$, where $C(A)$ – capacity of the cut A. The value of a flow in the network is not grater than the capacity of any cut.

<u>The Ford-and-Falkerson theorem</u>

Let's value of network flow φ_z and capacity of cut V such equality $\varphi_z = C(V)$ is true: then φ_z is the maximal flow that can be passed through the network and V has the minimal capacity from all the cuts in this network.

Then V is called **the minimal cut**.

An arc u u is called **saturated** if $\varphi(u) = C(u)$.

An arc u is called **empty** if $\varphi(u) = 0$.

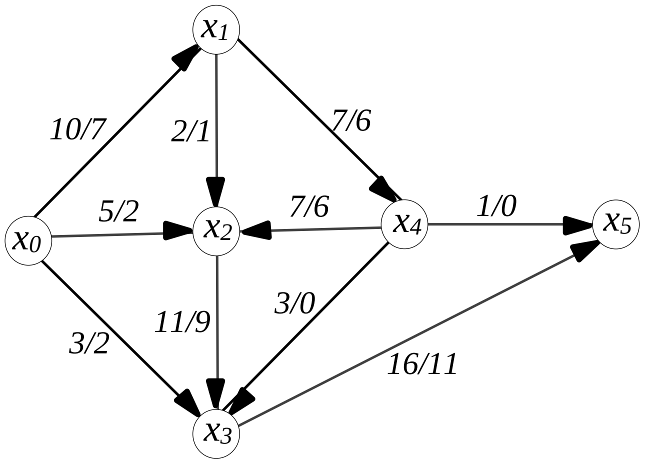
An arc u is called **busy** if $\varphi(u) > 0$.

<u>The algorithm consists of two stages:</u>

I – Find some complete flow;

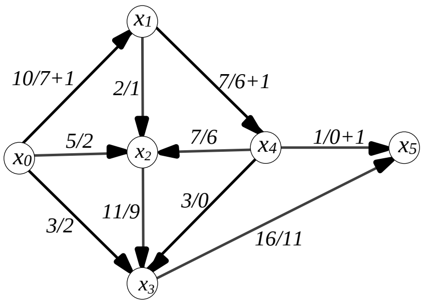
II – Evaluation maximal flow by given labels to the network vertices.

Let φ_z be some flow distributed of an arcs of a network. Find the complete flow when every path from x_0 to z contains at least one saturated arc.



Stage I (Ford-and-Falkerson algorithm)

Step 1. Let Δ is difference between the capacity of each arc and the flow in this arc.

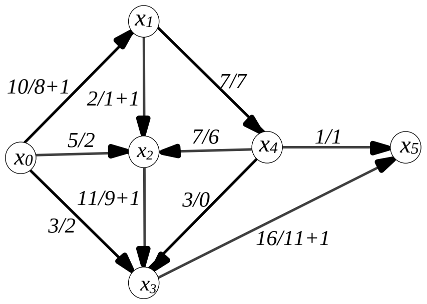


$\Delta = 1$. Arcs (x_1, x_4) and (x_4, z) are saturated.

$\mu_1 : x_0, x_1, x_4, z$; $\varphi_z = 12$

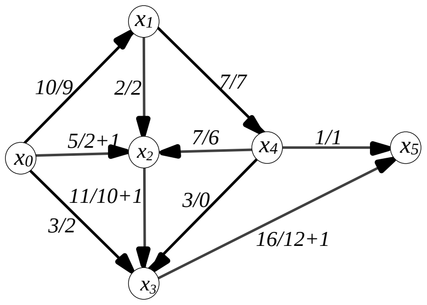
Step 2. $\Delta = 1$. Arc (x_1, x_2) is saturated.

$\mu_2 : x_0, x_1, x_2, x_3, z$; $\varphi_z = 13$



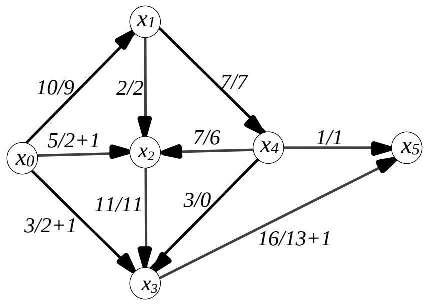
Step 3. $\Delta = 1$. Arc (x_2, x_3) is saturated.

$\mu_3 : x_0, x_2, x_3, z$; $\varphi_z = 14$



Step 4. $\Delta = 1$. Arc (x_0, x_3) is saturated.

$\mu_4 : x_0, x_3, z;$ $\varphi_z = 15$ corresponds to the complete flow, because each path from x_0 to z contains at least one saturated arc.



Stage II (The process of labeling)

The process of increasing the flow φ_u consists in labeling the vertices of the network by indexes, that indicate the path where can be changed. If such label can reach the sink z that the flow φ_u can be increased. After this vertices are labeled again.

Algorithm of labeling vertices

1. $x_0[0] \leftarrow$ label;

2. Labeling vertex.

If x_i has already a label then the label $[+i]$ assign to all unlabeling ajacence to x_0 vertices when the flow f_z pass through the unbusy arc and the direction of the flow f_z coincide with the direction of the considered arc. The label $[-i]$ assign to all unlabeling vertices that are connected with x_i by busy arc with head in x_i.

3. If the process of labeling vertices reaches the vertex z then go to the step 4 else the flow, received on to the previous step was the maximal value.

4. Exists $x_0 - z$ path μ with distinct vertices labeled (accurate to the sign) with the number of previous vertices. Path μ is built, starting form the vertex z.

The flow of the arc φ_u corresponds to:

$$\varphi'_u = \begin{cases} \varphi_u, \; if \;\; u \notin \mu \\ \varphi_u + 1, \; if \;\; the \; direction \; of \;\; the \; flow \; f(u) \; and \;\; the \; direction \; of \;\; arc \; u \; coincide \\ \varphi_u - 1, \; if \;\; the \; direction \; of \;\; the \; flow \; f(u) \; and \;\; the \; direction \; of \;\; arc \; u \; opposite \end{cases}$$

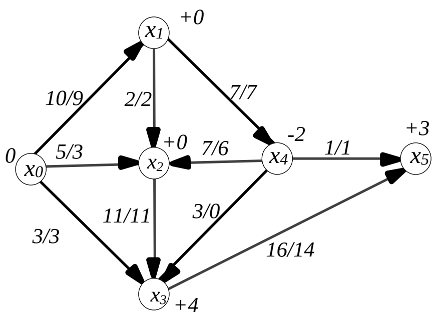
$\varphi'_u = \varphi_u + 1$ a new flow is received.

Go to the step 1.

The sink z was labeled.

There is a path : x_0, x_2, x_3, x_4, z where

the flow can be increased.



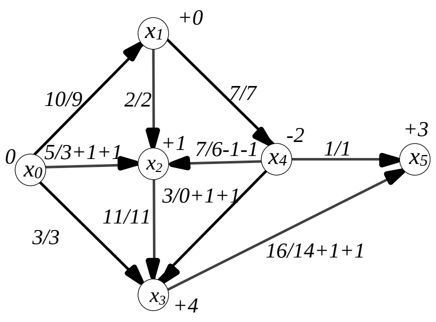
Increase the flow in such arcs:

$(x_0, x_2), (x_4, x_3), (x_3, z)$ by one.

Decrease the flow in the arc (x_4, x_2) by one.

There is a path $\mu_4 : x_0, x_2, x_4, x_3, z$

where the flow can be increased.



Permits the flow φ_u φu is increased in arcs

$(x_0, x_1), (x_1, x_2), (x_4, x_3), (x_3, z)$ and is decreased in (x_4, x_2).

The vertex z is not labeled then the flow received at the previous step was the maximal. $\varphi_z = 16$ – maximal flow.

Bibliography

1. Barker S. F. The Elements of Logic. New York: McGraw-Hill. 1989.
2. Birshoff G. and Maclane S. A Survey of Modern Algebra: 3rd edition, MacMillan. 2004.
3. Brualdi R. A. Introductory Combinations. Prentice-Hall. 1999.
4. Copi I. M. and C. Cohen Introduction to Logic (10th edition): Prentice-Hall. 1998.
5. Edgar W. J. The Elements of Logic. SRA: Chicago. 1989.
6. Gallier J. H. Logic for Computer Science . Harper & Row: New York. 1986.
7. Gould R. Graph Theory. Benjamim. 1988.
8. Tucker A. Applied Combinations. Wiley: New York. 1995.
9. West D., Introduction to Graph Theory. Prentice-Hall. Upper Saddle River. N. J. 2000.

ДЛЯ НОТАТОК

ДЛЯ НОТАТОК

ДЛЯ НОТАТОК

Навчальне видання

Кагадій Тетяна Station

Шпорта Анна Григорівна

ОСНОВИ ДИСКРЕТНОЇ МАТЕМАТИКИ

Навчальний посібник

(Англійською мовою)

Видано в редакції авторів
Коректор Ю.В. Рачковська

Підписано до друку 08.03.2023. Формат 30х42/4.
Папір офсетний. Ризографія. Ум. друк. арк. 3,5.
Обл.-вид. арк. 3,5. Тираж 50 пр. Зам. № 1

Підготовлено до друку та видрукувано
у Національному технічному університеті «Дніпровська політехніка».
Свідоцтво про внесення до Державного реєстру ДК № 1842 від 11.06.2004.
49005, м. Дніпро, просп. Д. Яворницького,19.