МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра информационных технологий и компьютерных систем

ДИСКРЕТНАЯ МАТЕМАТИКА И КОМПЬЮТЕРНАЯ ЛОГИКА. ТЕОРИЯ МНОЖЕСТВ, ТЕОРИЯ ОТНОШЕНИЙ, ТЕОРИЯ ГРАФОВ

Методические указания
к выполнению лабораторных работ по дисциплине
для студентов дневной формы обучения
по направлениям подготовки
09.03.01 «Информатика и вычислительная техника»,
09.03.04 «Программная инженерия»

Севастополь
СевГУ
2023

УДК 519.17+519.5/.6(076.5)
ББК 22.176я73
Д482

Р е ц е н з е н т :
В. В. Альчаков - канд. техн. наук, доцент кафедры
«Информатика и управление в технических системах»

Ответственный за выпуск:
А. А. Брюховецкий - канд. техн. наук, доцент, заведующий кафедрой
информационных технологий и компьютерных систем

Составители: О. В. Ченгарь, А. А. Малицкая

Д482 Дискретная математика и компьютерная логика. Теория множеств, теория отношений, теория графов : методические указания к выполнению лабораторных работ по дисциплине для студентов дневной формы обучения по направлениям подготовки 09.03.01 «Информатика и вычислительная техника», 09.03.04 «Программная инженерия» / Севастопольский государственный университет, Институт информационных технологий, кафедра информационных технологий и компьютерных систем ; сост.: О. В. Ченгарь, А. А. Малицкая. – Севастополь : СевГУ, 2023. – 163 с. – Текст : электронный.

В методических указаниях приводятся теоретические сведения и задания для выполнения лабораторных работ в процессе изучения учебной дисциплины «Дискретная математика и компьютерная логика».

УДК 519.17+519.5/.6(076.5)
ББК 22.176я73

Методические указания рассмотрены и утверждены на заседании кафедры «Информационные технологии и компьютерные системы», протокол № 5 от 04 февраля 2022 г.

СОДЕРЖАНИЕ

1. ЦЕЛЬ И ЗАДАЧИ ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

Дисциплина "Дискретная математика" является математическим базисом образования бакалавра в области компьютерной инженерии.

Она связана с дисциплинами "Высшая математика" и "Программирование", опирается на формируемые этими дисциплинами общую математическую и алгоритмическую культуру.

Дискретная математика используется в качестве математического аппарата большинства курсов по специальностям, связанных с проектированием, программированием и применением ЭВМ.

Освоение данной дисциплины является теоретической и практической базой для следующих дисциплин:

- Алгоритмизация и программирование;
- Компьютерная логика;
- Компьютерная схемотехника;
- Вычислительный практикум.

В результате выполнения лабораторных работ студенты должны приобрести следующие практические умения и навыки:

– формализации и алгоритмизации задачи;

– планирования, проведения и анализа результатов вычислительных экспериментов;

– оформления технической документации в соответствии с ГОСТ.

2. ТЕМАТИКА ЛАБОРАТОРНЫХ РАБОТ

Темы лабораторных заданий отвечают содержанию и задачам дисциплины «Дискретная математика и компьютерная логика» и непосредственно связаны с тематикой лабораторных занятий, проводимых в первом семестре в соответствии с рабочей программой дисциплины, а также научно-исследовательских и научно-методических работ, которые выполняются на кафедре.

Задания на лабораторные работы включают перечень задач для реализации, методы решения, варианты для определения входных данных, список рекомендуемой литературы.

Предлагаемый курс "Дискретная математика" в общем случае называется - математическая теория систем.

Дадим формальное определение системе.

Системой будем называть устройство преобразования информации.

Схематически систему можно представить следующим образом:

$$X(T) \Rightarrow \boxed{\mathbf{L}} \Rightarrow Y(T)$$

X(Y)-вектор входных информационных процессов, длины N, с компонентами Xi(T).

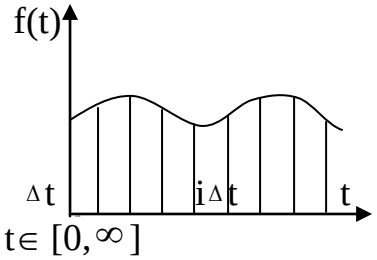
Y(T)-вектор выходных информационных процессов, длины N, с компонентами Yi(T).

T-множество моментов времени, в течение которого происходит работа системы.

Рассмотрим подробно информационные процессы, трансформируемые или передаваемые системы:

1.Аналоговые системы в непрерывном времени.

Представляются функцией f(t) [a;b] - область значений функции принадлежит интервалу [a;b].

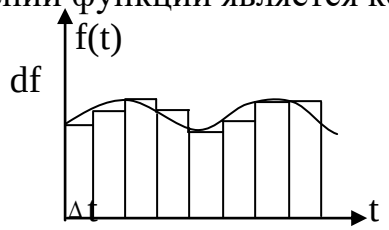


2.Аналоговые процессы в дискретном времени.

Дискретным временем назовем множество определенных моментов времени. Такие процессы представляются функцией от дискретного аргумента f(idt),где dt - интервал между выбранными временными отрезками (шаг дискретизации).

3.Квантованные процессы в непрерывном времени.

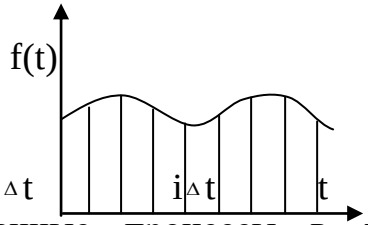
Эти процессы также представляются функцией f(t), но область значений функции является конечным множеством.



df - шаг квантования.

4.Квантовые процессы в дискретном времени.

Представлены функцией от дискретного аргумента idt и проквантованы по уровню.



Квантованные процессы в дискретном времени носят название цифровых процессов, потому что могут быть представлены последовательностью цифр им соответствующих, так, например если a=1, а b=6, то представляемый цифровой процесс задается рядом:

2.5 3.5 5 5 3.5 2 2 4 5 .

Символы соответствующие цифровому процессу называются алфавитом процесса.

В соответствие с выше указанным разделением информационных процессов их можно классифицировать по системам.

Так системы, работающие с информационными процессами 1 и 2, называются аналоговыми системами с непрерывным и дискретным временем соответственно. Системы, использующие процесс 3, называются квантованными системами с непрерывным временем. Системы, использующие процесс 4, называются цифровыми или дискретными системами.

Эта система очень важна для организации алгоритмов и структуры анализа, и будет, является основной целью изучения данного курса.

3. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

3.1 МНОЖЕСТВА

3.1.1 Основные понятия теории множеств. Диаграмма Эйлера - Венна

Теория множеств является логической основой современного математического аппарата.

Основатель теории множеств Георг Кантор определял множество следующим понятием: «под множеством понимают всякую совокупность определенных элементов, которое может быть связано с помощью некоторого закона."

Множество обозначают прописными буквами латинского алфавита.

Под **множеством А** будем понимать совокупность каких-либо объектов произвольной природы, обладающих некоторым общим признаком (рассматриваемых совместно).

Объекты, составляющие множество называются его **элементами** и обозначаются строчными латинскими буквами.

A={a,в,c}

Для того чтобы указать, что множество состоит из элементов x применяют, вот такую форму записи:

A={x}

Если нужно указать **характеристическое свойство** согласно которого объекты объединяются во множества применяется следующая запись: A={x | P(x)} где P(x) - характеристическое свойство (т.е. множество A состоит из элементов *x*, удовлетворяющих условию P(x)).

Пример:

$B=\{X:x^2-1=0\}$

Элементами множества B является множество корней уравнения $x^2-1=0$

Введем некоторые **основные понятия и обозначения**.

Некоторые множества имеют общепринятые имена:

N – множество натуральных чисел;

Z – множество целых чисел;

R – множество вещественных чисел;

$\emptyset$ - множество, не содержащее элементов, пустое.

Для того чтобы указать, что **x** есть элемент множества **A** пишут: $x \in A$

Чтобы указать, что **x** не принадлежит множеству **A** записывают таким образом: $x \notin A$

Если множества A и B совпадают, то пишут: A=B (это означает, что элементы этих множеств одни и те же).

$\subseteq$ - отношение включения.

$A \subseteq B$ - каждый элемент множества A является элементом множества B, A- **подмножество** B.

Пример:

B-множество всех студентов в аудитории.

A-множество всех студентов мужского пола.

|A| – **мощность** A – число элементов в конечном множестве A.

Множество всех подмножеств множества A называется **множеством-степенью** и обозначается P(A).

Если $|A|=n$, то $|P(A)|=2^n$.

Пример

Для множества A записать множество – степень P(A).

A{1,2,3}; P(A) = {∅,{1},{2},{3},{1,2},{1,3},{2,3},A}.

Условимся различать конечные и бесконечные множества.

Конечным множеством назовем множество, количество элементов которого может быть выражено конечным числом, причем неважно, что это за число. Главное, что оно существует. Примерами конечных множеств может служить количество рук человека, количество букв на странице конспекта, число букв во всех изданных книгах.

К **конечным** множествам мы также будем **относить и пустое** множество, не содержащее элементов.

К **бесконечным** множествам отнесем все множества, не являющиеся конечными. Примерами бесконечных множеств может служить множество всех целых чисел, множество точек на плоскости.

Конечные множества могут быть заданы простым перечислением его элементов.

Бесконечное множество может быть задано только указанием характеристического свойства элементов.

Геометрическое изображение множеств – **диаграмма Эйлера - Венна**.

Построение диаграммы заключается в изображении большого прямоугольника, представляющего **универсальное множество U (или E, или I)** (все рассматриваемые множества являются его подмножествами), а внутри него кругов (или каких-нибудь других замкнутых фигур), представляющих множества. Фигуры должны пересекаться в наиболее общем случае, требуемом в задаче, и должны быть соответствующим образом обозначены. Точки, лежащие внутри различных областей диаграммы, могут рассматриваться как элементы, соответствующих множеств.

3.1.2 Аксиомы теории множеств

Существует 6 изначальных аксиом теории множеств (все аксиомы мы будем сопровождать диаграммами Эйлера-Венна):

1. ***Аксиома существования:***

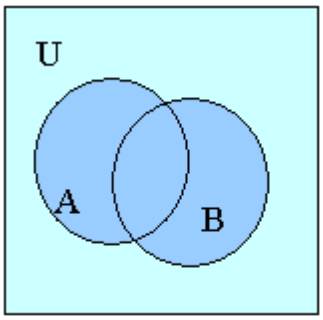
Существует хотя бы одно множество.

2. Аксиома эквивалентности:

Если множество A и B состоят из одних и тех же элементов, то они равны A=B

3. Аксиома объединения:

Для двух произвольных множеств **A** и **B** существует такое множество **C**, элементами которого является каждый элемент, содержащийся хотя бы в одном из этих множеств.



Аналитическое выражение для двух множеств:

$$A \cup B = \{x | x \in A \vee x \in B\}$$

Аксиома обобщается на случай нескольких исходных множеств и звучит так: "Для произвольных множеств A_i существует множество C, элементами которого является каждый элемент, содержащийся хотя бы в одном из этих множеств A_i".

Для операции объединения справедливы свойства:

1) A∪A=A
2) A∪U = U
3) A ∪ ∅ =A

Исходя из этих свойств бинарную операцию объединения обозначают следующим образом:

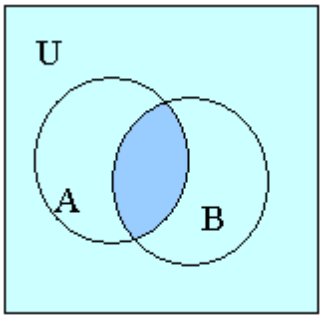
C=A+B

А множественную операцию обозначают:

$$M = \bigcup_{i=1}^{m} M_i$$

4. Аксиома пересечения:

Для двух произвольных множеств **A** и **B** существует множество **C** элементами которого является каждый элемент, принадлежащий как множеству A, так и множеству B.



Аналитическое выражение для двух множеств:

$$A \cap B = \{x | x \in A \,\&\, x \in B\}$$

Для операции пересечения справедливы следующие соотношения:

1) A ∩ A = A
2) A ∩ U = U
3) A ∩ ∅ =∅

Обобщенная аксиома на случай нескольких исходных множеств:

"Для произвольных множеств A_i существует множество C, элементами которого является каждый элемент, принадлежащий множествам A_i

одновременно".

Далее бинарная операция будет обозначаться:

C=A*B

А множественная:

$M= \bigcap_{i=0}^{m} M_i$

5. Аксиома универсального множества:

"Для произвольной группы множеств A_i всегда можно выбрать такое множество U, что $A_i \subset U$".

Множество U назовем единичным множеством (универсальным).

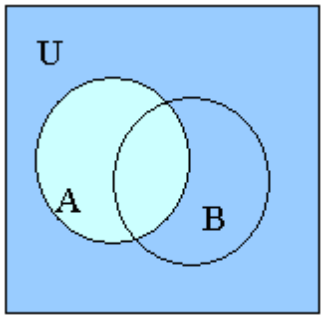
6. Аксиома пустого множества:

Всегда существует пустое множество - ∅, которому не принадлежит ни один элемент (иначе нулевое множество).

3.1.3 Дополнительные операции над множествами

1. Операция отрицания (дополнения):

Для произвольного множества А существует дополнение к единичному множеству, обозначаемое $\overline{A}$ (не А).



Аналитическое выражение для двух множеств:

$$\overline{A} = \{x | x \in U \,\&\, x \notin A\}$$

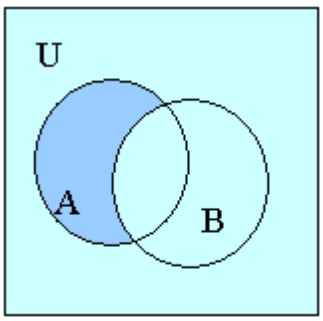
Следствие:

1) $A \cup \bar{A} = U$

2) $A \cap \bar{A} = \emptyset$

2. Разность между множествами:

Для произвольных множеств А, В существует множество С, определяемое как



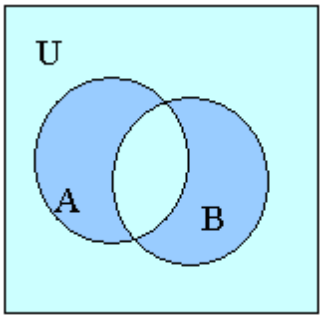
Аналитическое выражение для двух множеств:

$$A \backslash B = \{x | x \in A \,\&\, x \notin B\}$$

Бинарная операция:

C=A-B

3. *Симметрическая разность множеств A,B.*

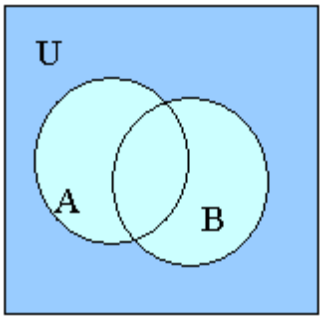


Аналитическое выражение для двух множеств:

$$A\Delta B = A \oplus B = (A \backslash B) \cup (B \backslash A) =$$
$$= \{x | x \in A \,\&\, x \in B \,\&\, x \in A \cap B\}$$

Бинарная операция:
C=(A-B)+(B-A)=A Δ B

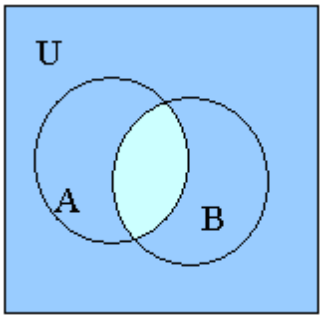
4. *Стрелка Пирса A и B*



Аналитическое выражение для двух множеств:

$$A \downarrow B = \overline{\mathrm{A} \cup \mathrm{B}} =$$
$$= \{x | x \in U \,\&\, x \notin A \,\&\, x \notin B\}$$

5. *Штрих Шеффера A и B*



Аналитическое выражение для двух множеств:

$$A | B = \overline{\mathrm{A} \cap \mathrm{B}} =$$
$$= \{x | x \in U \,\&\, x \notin A \cap B\}$$

3.1.4 Основные законы алгебры множеств

1. *Закон коммутативности*

A ∪ B = B ∪ A
A ∩ B = B ∩ A

2. *Закон ассоциативности*

A∪(B∪C)=(A∪B)∪C
A∩(B∩C)=(A∩B)∩C

3. *Закон дистрибутивности*

- (A∪B)∩C=(A∩C)∪(B∩C)

Доказательство:

Рассмотрим выражение в левой части:
M=(A∪B)∩C

Если $x \in M$, то значит, что $x \in C$ и одновременно $\in A$ или $\in B$

Когда $x \in A$, $x \in (A \cap C)$, а когда $x \in B$ $x \in (B \cap C)$

Объединение этих выражений даёт правую часть.

- $(A \cap B) \cup C = (A \cup C) \cap (B \cup C)$

Доказательство:

Возьмем правую часть равенства. Согласно закону ассоциативности, раскроем скобки и получим:

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

(упростили, используя закон поглощения). Из записи закона ассоциативности и закона дистрибутивности видно, что один закон можно получить из другого, заменив знаки "$\cup$" и "$\cap$", следовательно, законы двойственны.

4. Закон поглощения

Если A содержится в B, то $A \cup B = B$ и $A \cap B = A$

- $A \cup (A \cap B) = A$

Следствие:

Если B=1, то $A \cup A = A$

- $A \cap (A \cup B) = A$

Свойство степени:

Если множество пересекается с самим собой, то из определения пересечения следует $A \cap A = A$

5. Законы де Моргана.

Эти законы позволяют выразить законы объединения и пересечения друг через друга с использованием операции дополнения:

а) $A \cup B = \overline{\bar{A} \cap \bar{B}}$

Доказательство:

Обозначим через M: $M = A \cup B$ и $\overline{M} = \overline{A} \cap \overline{B}$. Если теперь объединение M и $\overline{M}$ даст единичное множество, то закон будет доказан.

$M \cup \overline{M} = A \cup B \cup \overline{A} \cap \overline{B} = A \cup (B \cup \overline{A}) \cap (B \cup \overline{B})$.

Используя определение дополнения получим: $M \cup \overline{M} = A \cup B \cup \overline{A} = 1 \cup B = 1 = U$

б) $A \cap B = \overline{\bar{A} \cup \bar{B}}$

Доказательство:

Обозначим через M: $M = A \cap B$ и $\overline{M} = \overline{A} \cup \overline{B}$. Если теперь объединение M и $\overline{M}$ даст единичное множество, то закон будет доказан.

$M \cup \overline{M} = A \cap B \cup \overline{A} \cap \overline{B} = (\overline{A} \cup A) \cap (B \cup \overline{A}) \cup \overline{B} = B \cup \overline{A} \cup \overline{B} = 1 \cup \overline{A} = 1 = U$

Законы де Моргана так же являются двойственными.

$A \cap B = AB$.

3.1.5 Выводы по разделу

1) $A \subset A$
2) Если $A \subset B$ и $B \subset A$, то $A = B$
3) Если $A \subset B$ и $B \subset C$, то $A \subset C$
4) $\varnothing \subset A$
5) $A \subset U$
6) $A \cup B = B \cup A$
7) $A \cap B = B \cap A$
8) $A \cup (B \cup C) = (A \cup B) \cup C$
9) $A \cap (B \cap C) = (A \cap B) \cap C$
10) $A \cup A = A$
11) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
12) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
13) $A \cup \varnothing = A$
14) $A \cup U = U$
15) $A \cap U = A$
16) $A \cap A = A$
17) $A \cap \varnothing = \varnothing$
18) Если $A \subset B$, то $A \cup B = B$, $A \cap B = A$
19) $A \cup \overline{A} = U$
20) $A \cap \overline{A} = \varnothing$
21) $\overline{\varnothing} = U$
22) $\bar{U} = \varnothing$
23) $\overline{\overline{A}} = A$
24) Если $A \subset B$, то $\overline{B} \subset \overline{A}$
25) $(\overline{A \cup B}) = \overline{A} \cap \overline{B}$
26) $(\overline{A \cap B}) = \overline{A} \cup \overline{B}$

3.2 БИНАРНЫЕ ОТНОШЕНИЯ

3.2.1 Основные определения

Прямым(декартовым) произведением множеств *A и B* называется множество *M* всех пар *(a,b)*, таких, что *a* принадлежит *A*, а *b* принадлежит *B*:

$$M = A \times B = \{(a, b) \mid a \in A, b \in B\}$$

Причем:

$$A \times B \neq B \times A$$

Пример

$$A = \{1,2,3\}\ B = \{8,9\}$$
$$A \times B = \{(1,8), (1,9), (2,8), (2,9), (3,8), (3,9)\}$$
$$B \times A = \{(8,1), (8,2), (8,3), (9,1), (9,2), (9,3)\}$$

Бинарным (двуместным) отношением между элементами a, b множеств *A* и *B* называется любое подмножество *R* множества $A \times B$.

Если *A=B*, то отношение называется **бинарным отношением на A**.

Вместо *(a,b)* часто записывают *aRb*.

Областью определения бинарного отношения R называется множество

$$D_R = \{a | \exists b \in B : (a, b) \in R\}$$

Областью значений бинарного отношения R называется множество:

$$Q_R = \{b | \exists a \in A : (a, b) \in R\}$$

3.2.2 Способы задания бинарных отношений

1. перечислением пар, для которых это отношение выполняется;

2. матрицей *C*;

$$C_{ij} = \begin{cases} 1, (a_i, b_j) \in R \\ 0, (a_i, b_j) \notin R \end{cases}$$

Пример

A={1,2,3,4} $a, b \in A$

R - < («быть строго меньше»),

aRb – «*a* строго меньше *b*».

Задать это отношение разными способами:

1. перечисление пар, для которых это отношение выполняется:

$$R = \{(1,2), (1,3), (1,4), (2,3), (2,4), (3,4)\}$$

2. матрица *C*:

R	1	2	3	4
1	0	1	1	1
2	0	0	1	1
3	0	0	0	1
4	0	0	0	0

3.2.3 Свойства бинарных отношений

Бинарное отношение *R* на множестве *A* называется

рефлексивным, если

$$(a, a) \in R \qquad \forall a \in A,$$

антирефлексивным, если

$$(a, a) \notin R \qquad \forall a \in A.$$

Если хотя бы для одного $a \in A$ пара $(a, a) \notin R$, то отношение называют **нерефлексивным**.

Бинарное отношение R на множестве A называется **симметричным**, если из

$$(a, b) \in R \Rightarrow (b, a) \in R \qquad \forall a, b \in A \ ,$$

асимметричным, если это условие не выполняется:

$$(a, b) \in R \Rightarrow (b, a) \notin R \qquad \forall a, b \in A$$

антисимметричным (выполнение отношений *aRb и bRa* влечёт $a = b$, или, то же самое, выполнение отношений aRb и bRa возможно только для равных a и b**.)**, если из

$$(a, b) \in R \ \ \& \ \ (b, a) \in R \qquad \Rightarrow \qquad a = b \ .$$

Причем: **антисимметричность отношения не исключает симметричности!**

Бинарное отношение R на множестве A называется **транзитивным**, если из

$$(a,b) \in R \ \ \& \ \ (b,c) \in R \qquad \Rightarrow \qquad (a,c) \in R \ .$$

Рефлексивное, симметричное и транзитивное бинарное отношение R называется **отношением эквивалентности**.

Примеры

1. Рассмотрим бинарное отношение R – «быть строго меньше», aRb – «a строго меньше b» на множестве натуральных чисел.

$$R = \{(a,b) | a < b\}$$

Это отношение антирефлексивно ($a<a$ не выполняется для всех a принадлежащих N), асимметрично (a<b и b<a не выполняется) и транзитивно (из $a<b$ и $b<c$ следует, что $a<c$).

2. Рассмотрим бинарное отношение R – «жить в одном городе», aRb – «a живет в одном городе с b» на множестве всех людей.

Это отношение рефлексивно («a живет в одном городе с a», симметрично (из «a живет в одном городе с b» следует, что «b живет в одном городе с a»), транзитивно (из «a живет в одном городе с b» и «b живет в одном городе с c» следует, что «a живет в одном городе с c»), следовательно это отношение эквивалентности.

Отношение толлерантности – рефлексивное, симметричное, но нетранзитивное бинарное отношение R.

Пусть R – отношение эквивалентности на AxA, т.е. R – рефлексивно, симметрично, транзитивно.

Классом эквивалентности, порожденным элементом a, называется подмножество A, состоящее из b принадлежащих A, т.ч., выполняется aRb.

Класс эквивалентности, порожденный элементом a, обозначается:

$$[a] = \{b \mid b \in A \ \ \& \ \ aRb\}.$$

Примеры

1. R – отношение равенства на множестве целых чисел Z.

$$[a] = \{a \mid a \in Z \ \ \& \ \ aRa \ \ (a = a)\}.$$

2. R – отношение принадлежности к одной группе на множестве студентов факультета.

Класс эквивалентности – множество студентов одной группы.

Утверждение 1

Пусть R – отношение эквивалентности на X.

Тогда, если:

1) $x \in X \ \Rightarrow \ x \in [x]$,

2) $x, y \in X \ \ \& \ \ xRy \ \Rightarrow \ [x] = [y]$,

класс эквивалентности порождается любым своим элементом.

Доказательство

1) R – отношение эквивалентности, следовательно, оно рефлексивно, т.е. xRx, следовательно, $x \in [x]$.

2) Пусть $z \in [y]$,следовательно, *yRz, xRy*.].

Из этого, в силу транзитивности, следует, *xRz*, т.е. $z \in [x]$.

Отсюда $[y] \subseteq [x]$.

Аналогично в силу симметричности *R* можно показать, что $[x] \subseteq [y]$, а значит, [x]=[y].

Разбиением множества *X* называется совокупность попарно непересекающихся подмножеств множества *X* таких, что каждый элемент множества *X* принадлежит одному и только одному из этих подмножеств.

Пример

Привести примеры разбиения множества X={1,2,3,4,5,6}.

1) {{1,2,3},{4},{5,6}};
2) {{1},{2,3},{4,5},{6}};
3) {{1,2,3},{4,5,6}}

и т.п.

Утверждение 2

Всякое разбиение множества *X* определяет на *X* отношение эквивалентности *R*: *xRy* тогда и только тогда, когда *x* и *y* принадлежат одному подмножеству разбиения.

Доказательство

Рефлексивность и симметричность *R* очевидны. Пусть теперь *xRy* и *yRz*. Тогда $x, y \in X_1, \quad y, z \in X_2,$

где X_1, X_2- подмножества из разбиения *X*.

Поскольку $y \in X_1 \ \& \ y \in X_2 \ \Rightarrow \ X_1 = X_2$.

Следовательно, $x, z \in X_1 \ \Rightarrow \ xRz$.

Утверждение 3

Всякое отношение эквивалентности *R* определяет разбиение множества *X* на классы эквивалентности относительно этого отношения.

Доказательство

Докажем, что совокупность классов эквивалентности определяет разбиение множества *X*.

В силу утверждения 1 $x \in [x]$, а, следовательно, каждый элемент множества *X* принадлежит некоторому классу эквивалентности. Из утверждения 1 вытекает также, что два класса эквивалентности либо не пересекаются, либо совпадают, так как, если $z \in [x] \ \& z \in [y] \ \Rightarrow \ xRz$

Откуда *[x]=[y] & yRz*. Следовательно, *[y]=[z]*. И окончательно: *[x]=[y]*.

Совокупность классов эквивалентности элементов множества *X* по отношению эквивалентности *R* называется **фактор – множеством** множества *X* по отношению *R* и обозначается *X\R*.

Рассмотрим еще несколько типов специальных бинарных отношений.

Рефлексивное, антисимметричное и транзитивное отношение называется **отношением нестрогого порядка на множестве** *X* и обозначается символом $\preceq$.

Антирефлексивное, антисимметричное и транзитивное отношение называется **отношением строгого порядка на множестве** *X* и обозначается символом $\prec$.

Оба эти отношения называются **отношениями порядка**.

Элементы *x,y* принадлежащие *X* **сравнимы по отношению порядка** *R на X*, если выполняется *xRy* или *yRx*.

Множество *X* , на котором задано отношение порядка, может быть:

1) **полностью упорядоченным множеством**, если любые два элемента из *X* сравнимы по отношению порядка. В таком случае говорят, что отношение *R* задает полный порядок на множестве *X*.

Например, отношение «быть не старше» задает полный порядок на множестве людей;

2) **частично упорядоченным множеством** – в противном случае. При этом говорят, что отношение *R* задает на множестве *X* частичный порядок.

Например, отношение «быть начальником» задает на множестве сотрудников организации частичный порядок, так как, например, для пары сотрудников это отношение может не выполняться: они не сравнимы по данному отношению.

Примеры

Какой порядок на множестве задают отношения:

1. «не больше», «не меньше», «больше», «меньше» на множествах натуральных *N* и действительных *R* чисел.

Отношения нестрогого порядка ≥ и ≤, а также строгого порядка > и < полностью упорядочивают множества *N* и *R*.

2. подчиненности на предприятии.

Отношение подчиненности на предприятии задает строгий частичный порядок. В нем несравнимыми являются, например, сотрудники разных звеньев одного уровня.

3.2.4 Операции над бинарными отношениями

Пусть бинарные отношения *R, R1, R2* – подмножества *A*x*B* (может быть *A=B*).

Символы: &- «и», V- «или», ∃- «существует».

1. *Объединение*:
$$R_1 \cup R_2 = \{(a,b) | (a,b) \in R_1 \vee (a,b) \in R_2\}.$$
2. *Пересечение*:
$$R_1 \cap R_2 = \{(a,b) | (a,b) \in R_1 \& (a,b) \in R_2\}.$$
3. *Разность*:
$$R_1 \backslash R_2 = \{(a,b) | (a,b) \in R_1 \& (a,b) \notin R_2\}.$$
4. *Дополнение*:
$$\bar{R} = (A \times B) \backslash R.$$
5. *Обратное отношение для R*:
$$R^{-1} = \{(a,b) | (b,a) \in R\}.$$
6. *Произведение отношений (композиция)*:
$$R_1 \circ R_2 = \{(a,c) | \exists b \in B : (a,b) \in R_1 \& (b,c) \in R_2\},$$
$$R_1 \subseteq A \times B \quad \& \quad R_2 \subseteq B \times C.$$

Свойства операций

1. $(R^{-1})^{-1} = R$
2. $(R_1 \cup R_2)^{-1} = {R_1}^{-1} \cup {R_2}^{-1}$
3. $(R_1 \cap R_2)^{-1} = {R_1}^{-1} \cap {R_2}^{-1}$
4. $(R_1 \circ R_2)^{-1} = {R_1}^{-1} \circ {R_2}^{-1}$

Примеры

1. Пусть *R1*, *R2* – отношения, заданные на A={1,2,3,4,5,6,7,8,9},

$$R_1 = \{(a,b) | b = a + 2;\ \ a,b \in A\},$$
$$R_2 = \{(a,b) | b = a^2;\ \ a,b \in A\}.$$

Определить отношения

$R_1 \cup R_2$, $R_1 \cap R_2$, $R_1 \backslash R_2$, $R_1 \circ R_2$, $R_2 \circ R_1$, $\overline{R_1}$, R_1^{-1}

Зададим отношения перечислением пар:

$$R_1 = \{(1,3),(2,4),(3,5),(4,6),(5,7),(6,8),(7,9)\},$$
$$R_2 = \{(1,1),(2,4),(3,9)\},$$

тогда:

$$R_1 \cup R_2 = \{(1,1),(1,3),(2,4),(3,5),(3,9),(4,6),(5,7),(6,8),(7,9)\};$$
$$R_1 \cap R_2 = \{(2,4)\};$$
$$R_1 \setminus R_2 = \{(1,3),(3,5),(4,6),(5,7),(6,8),(7,9)\}$$
$$R_1 \circ R_2 = \{(a,c) | b = a + 2; b^2 = c;\ \ a,b,c \in A\} =$$
$$= [(a+2)^2 = c;\ \ a,c \in A] = \{(1,9)\};$$
$$R_2 \circ R_1 = \{(a,c) | a^2 = b;\ \ c = b + 2;\ \ a,b,c \in A\} =$$
$$= [a^2 = c - 2;\ \ a,c \in A] = \{(1,3),(2,6)\};$$

$$\overline{R_1} = \left\{ \begin{array}{c} (1,1),(1,2),(1,4),(1,5),(1,6),(1,7),(1,8),(1,9), \\ (2,1),(2,2),(2,3),(2,5),(2,6),(2,7),(2,8),(2,9), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ (7,1),(7,2),(7,3),(7,4),(7,5),(7,6),(7,7),(7,8), \\ (8,1),(8,2),(8,3),(8,4),(8,5),(8,6),(8,7),(8,8),(8,9), \\ (9,1),(9,2),(9,3),(9,4),(9,5),(9,6),(9,7),(9,8),(9,9) \end{array} \right\}$$

$$R_1^{-1} = \{(b,a) | b = a + 2;\ \ a,b \in A\} =$$
$$\{(3,1),(4,2),(5,3),(6,4)(7,5),(8,6),(9,7)\}.$$

1. *R*- «быть начальником».

Определить: $\bar{R}$, R^{-1}

$\bar{R}$ - «не быть начальником»

R^{-1} – «быть подчиненным»

3.2.5 Операции покрытия

$$R = \ni$$
$$X = \{\{y_1, y_3\}, \{y_1, y_4\}, \{y_2, y_4\}\}$$
$$Y = \{y_1, y_2, y_3, y_4\}$$

∋	y_1	y_2	y_3	y_4
$\{y_1, y_3\}$	1	0	1	0
$\{y_1, y_4\}$	1	0	0	1
$\{y_2, y_4\}$	0	1	0	1

3.3 ОПТИМИЗАЦИОННЫЕ ЗАДАЧИ НА КОНЕЧНЫХ МНОЖЕСТВАХ

3.3.1 Задача о покрытии. Постановка задачи

Задача о покрытии – это математическая модель большого числа различных оптимизационных задач дискретной математики.

Рассмотрим в качестве ***<u>примера</u>*** задачу «О полном собрании сочинений».

Некоторый писатель выпустил множество $\mathrm{A} = \{\mathrm{A}_1, \dots, A_m\}$ сборников, содержащих в совокупности все множество $B = \{B, \dots, B_n\}$ его сочинений. Каждый сборник A_i содержит некоторое подмножество сочинений из B $(A_i \subseteq B)$ и имеет некоторую цену a_i. Необходимо найти такое множество $P = \{\mathrm{A}_{i_1}, \dots, \mathrm{A}_{i_l}\}$, $(i_j \in \{1, \dots, m\}, l \leq m)$ сборников, чтобы в их совокупности содержались все сочинения данного автора (3.1):

$$\bigcup_{j=1}^{l} A_{i_j} = B \qquad (3.1)$$

при этом цена $S = \sum_{j=1}^{l} a_{i_j}$ такого полного собрания сочинений была наименьшей, либо количество l сборников было минимальным.

<u>Пример:</u>

$B = \{b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad b_6 \quad b_7 \quad b_8 \quad b_9\}$

$\mathrm{A}_1 = \{\mathrm{b}_1 \quad \mathrm{b}_3 \quad \mathrm{b}_6 \quad \mathrm{b}_7 \quad\} \; a_1 = 1;$

$\mathrm{A}_2 = \{\mathrm{b}_1 \quad \mathrm{b}_2 \quad \mathrm{b}_4 \quad \mathrm{b}_8 \quad\} \; a_2 = 2{,}5;$

$\mathrm{A}_3 = \{\quad \mathrm{b}_2 \quad \mathrm{b}_5 \quad \mathrm{b}_7 \quad\} \; a_3 = 2;$

$\mathrm{A}_4 = \{\quad \mathrm{b}_3 \quad \mathrm{b}_8 \quad \mathrm{b}_9\} \; a_4 = 1{,}5;$

$\mathrm{A}_5 = \{\quad \mathrm{b}_4 \quad \mathrm{b}_5 \quad \mathrm{b}_8 \quad \mathrm{b}_9\} \; a_5 = 3;$

$\mathrm{A}_6 = \{\quad \mathrm{b}_3 \quad \mathrm{b}_5 \quad \mathrm{b}_6 \quad \mathrm{b}_7 \quad \mathrm{b}_9\} \; a_6 = 7;$

$\mathrm{A}_7 = \{\quad \mathrm{b}_2 \quad \mathrm{b}_4 \quad \mathrm{b}_6 \quad\} \; a_7 = 1.$

Множество $P_i = \{\mathrm{A}_{i_1}, \dots, \mathrm{A}_{i_l}\}$, удовлетворяющее условию (1.1), является возможным решением задачи. Рассмотрим некоторые из таких решений:

$P_1 = \{\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_5\}, \quad \mathrm{A}_1 \cup \mathrm{A}_2 \cup \mathrm{A}_5 = B,$

цена S_1=1+2,5+3=6,5; l_1=3

$P_2 = \{\mathrm{A}_2, \mathrm{A}_6\}, \quad \mathrm{A}_2 \cup \mathrm{A}_6 = B,$

цена S_2=2,5+7=9,5; l_2=2

$P_3 = \{\mathrm{A}_1, \mathrm{A}_3, \mathrm{A}_4, \mathrm{A}_7\}, \quad \mathrm{A}_1 \cup \mathrm{A}_3 \cup \mathrm{A}_4 \cup \mathrm{A}_7 = B,$

цена S_3=1+2+1,5+1=5,5;l_3=4

Можно построить еще много других решений. Но не всякое подмножество из A – решение. Например, подмножество $\{A_1, A_2\} \subseteq A$ сборников не является полным сборником сочинений, поскольку не содержит сочинений b_5 и b_9.

Условие (3.1) – необходимое условие построения решения; условия 1) $S \to min$, или 2) $l \to min$ служат критериями, по которым из возможных решений P_1, P_1, ... выбираются наилучшие.

В даном примере из рассмотренных по критерию 1) выбирается P_2, по критерию 2) - P_2. Так ка мы не рассматривали другие возможные решения, то не можем гарантировать, что найденные решения – наилучшие. Действительно, относительно P_2 для данного примера можно подтвердить оптимальность решения по критерию 2) дополнительными рассуждениями: ни один из сборников $\mathrm{A}_1, \dots, \mathrm{A}_7$ не содержит все сочинения автора, таким образом $l \geq 2$; так как для P_2 величина $l = 2$, то это решение содержит минимально возможное количество сборников.

Задача о покрытии на языке теории множеств.

Абстрагируясь от возможных интерпретаций, сформулируем задачу о покрытии на языке теории множеств.

Пусть $B = \{b_1, \dots, b_n\}$ – **опорное множество**. Имеется множество $\mathrm{A} = \{\mathrm{A}_1, \dots, A_m\}$ подмножеств множества B $(A_i \subseteq B)$, $\bigcup_{i=1}^{m} A_i = B$. Каждому подмножеству A_i сопоставлена цена a_i. Множество $P = \{\mathrm{A}_{i_1}, \dots, \mathrm{A}_{i_l}\}$, $(l \leq m)$ называется решением задачи о покрытии или просто **покрытием**, если выполнено условие (3.1).

Объяснением термина «покрытие» служит то, что совокупность множеств $\mathrm{A}_{i_1}, \dots, \mathrm{A}_{i_l}$ содержит все элементы множества B, т.е. покрывает множество B.

Теорема Если P – покрытие, то $P' \supseteq P$ – тоже покрытие, т.е. множество всех возможных покрытий вогнуто.

Определения:

Покрытие P называется **безызбыточным**, если при удалении из него хотя бы одного элемента оно перестает быть покрытием. Иначе – покрытие **избыточно**.

Покрытие P называется **минимальным**, если его цена $S = \sum_{j=1}^{l} a_{i_j}$ – наименьшая среди всех покрытий данной задачи.

Покрытие P называется **кратчайшим**, если количество l сборников – наименьшее среди всех покрытий задачи.

Теорема Минимальные и кратчайшие покрытия – безызбыточны.

3.3.2 Таблица покрытий

Таблица покрытий T весьма удобна для представления исходных данных в задаче о покрытиях. Эта таблица – матрица T отношения принадлежности элементов множеств $A_i \in A$ опорному множеству B. Столбцы матрицы T сопоставлены элементам множества B, строки – элементам множества A:

$$T_{ij} = \begin{cases} 1, \text{если } b_j \in B \text{ и } b_j \in A_i \\ \emptyset, \text{если } b_j \in B \text{ и } b_j \notin A_i \end{cases}$$

Нули в матрице T не проставляются. Для рассмотренного примера (задача «О полном собрании сочинений») матрица имеет следующий вид:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	a_i
A_1	1		1			1	1			1
A_2	1	1		1				1		2,5
A_3		1			1		1			2
A_4			1					1	1	1,5
A_5				1	1			1	1	3
A_6			1		1	1	1		1	7
A_7		1		1		1				1

На языке таблицы покрытий задача о покрытии формулируется как задача построения таких подмножеств P_i строк таблицы покрытий, чтобы в совокупности строк, входящих в P_i, во всех столбцах были едины. Такое подмножество строк называется **покрытием**.

В данном ***примере*** $P_1 = \{A_1, A_2, A_5\}$ – покрытие, в чем легко убедиться по следующей таблице:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9
A_1	1		1			1	1		
A_2	1	1		1				1	
A_5				1	1			1	1

По крайней мере хотя бы одна единица есть в каждом столбце. На этом примере наглядно представлена информация на языке таблицы покрытий. Эта таблица также позволяет подтвердить выполнение необходимого условия (3.1). таблица покрытий позволяет упростить процесс поиска покрытий, поэтому в дальнейшем используется именно это представление задачи о покрытии.

3.3.3 Решение задачи методом полного перебора

Очевидным способом построения всех возможных покрытий служит метод перебора всех подмножеств строк таблицы покрытий: пустое подмножество строк, подмножества из одной строки, из двух строк и т.д., из всех строк. Приходится рассматривать 2^n подмножеств. Их можно представить в виде следующего перечисления подмножеств:

$$\left\{\begin{matrix}\emptyset, \{A_1\}, \dots, \{A_m\}, \{A_1, A_2\}, \dots, \{A_1, A_m\}, \{A_2, A_3\}, \dots, \{A_2, A_m\}, \{A_3, A_4\}, \dots, \\ \{A_{m-1}, A_m\}, \{A_1, A_2, A_3\}, \dots, \{A_1, \dots, A_m\} = D\end{matrix}\right\} \quad (3.2)$$

Например, в рассмотренном примере (3.1) 2^7=128 подмножеств.

Пример:

	b_1	b_2	b_3	b_4	b_5
A_1	1		1		1
A_2	1	1		1	
A_3	1			1	1
A_4		1	1		

Множество $\{D\}$ подмножеств $\{D_i\}$, которые строятся при полном переборе, будет следующим:

$$\{\emptyset, \{A_1\}\{A_2\}, \{A_3\}, \{A_4\}, \{A_1, A_2\}, \{A_1, A_3\}, \{A_1, A_4\}, \{A_2, A_3\}, \{A_2, A_4\}, \{A_3, A_4\},$$
$$\{A_1, A_2, A_3\}, \{A_1, A_2, A_4\}, \{A_1, A_3, A_4\}, \{A_2, A_3, A_4\}, \{A_1, A_2, A_3, A_4\}\} = D$$

Каждое $\{D_i \in D\}$ проверяется, является ли оно покрытием ($\emptyset, A_1, A_2, A_3, A_4$ очевидно покрытиями быть не могут).

$\{A_1, A_2\}$ – покрытие (совокупность этих двух строк содержит единицы во всех столбцах таблицы покрытий);

$\{A_1, A_3\}$ – не содержит единицы в столбце b_2;

$\{A_1, A_4\}$ – не содержит единицы в столбце b_4;

$\{A_2, A_3\}$ – не содержит единицы в столбце b_3;

$\{A_2, A_4\}$ – не содержит единицы в столбце b_5;

$\{A_3, A_4\}$ – покрытие;

$\{A_1, A_2, A_3\}$ – избыточное покрытие (так как поглощает $\{A_1, A_2\}$);

$\{A_1, A_2, A_4\}$ – избыточное покрытие;

$\{A_1, A_3, A_4\}$ – избыточное покрытие;

$\{A_2, A_3, A_4\}$ – избыточное покрытие;

$\{A_1, A_2, A_3, A_4\}$ – избыточное покрытие.

Очевидно, что для более сложных случаев представлять множество всех подмножеств будет весьма трудно. Для проверки очередного подмножества на соответствие условию (3.1) другие подмножества не нужны, поэтому достаточно на каждом шаге решения иметь одно подмножество и получать после рассмотрения очередного – другое подмножество, которое подлежит рассмотрению.

Алгоритм последовательного построения подмножеств *должен*:

– гарантировать последовательное построение всех возможных подмножеств, начиная с некоторого начального подмножества;

– избегать построения уже построенных подмножеств;

– быть простым и наглядным;

– иметь критерий останова (окончания перебора).

Таких алгоритмов достаточно много, но все они основаны на упорядочивании перебора подмножеств. Один из таких алгоритмов представлен формулой (3.2).

Другой алгоритм полного перебора является более удобным для дальнейшего упрощения процесса решения.

Будем считать, что подмножества A_i упорядочены следующим образом: $A_1, \dots, A_m$. Основная идея алгоритма: сначала строятся все подмножества D_i, содержащие A_1, затем – содержащие A_2, но не содержащие A_1. Если построено подмножество D_i, то за ним строятся подмножества D_{i+j}, целиком содержащие D_i ($D_i \subseteq D_{i+j}$.

Сформулируем **алгоритм полного перебора**:

0. Текущее подмножество $D = \{A_1\}, i = 0$.

1. $i = i + 1$. Запоминаем множество D как очередное построенное подмножество D_i.

2. Находим наибольший номер j элемента $A_j \in D$. Если $j \neq n$, то переходим к п.3 данного алгоритма. Если $j = n$, то удаляем A_n из D и если

$D = \emptyset$, то заканчиваем построение, если $D \neq \emptyset$, то находим наибольший номер j элемента $A_j \in D$ и удаляем A_j из D.

3. $j = j + 1$. Вводим в D элемент A_j. Переходим к п.1.

Выполнение алгоритма рассмотрим на ***примере3.2***. строится такая последовательность подмножества D_i:

$D_1 = \{A_1\}$,
$D_2 = \{A_1, A_2\}$,
$D_3 = \{A_1, A_2, A_3\}$,
$D_4 = \{A_1, A_2, A_3, A_4\}$,
$D_5 = \{A_1, A_2, A_4\}$,
$D_6 = \{A_1, A_3\}$,
$D_7 = \{A_1, A_3, A_4\}$,
$D_8 = \{A_1, A_4\}$,
$D_9 = \{A_2\}$,
$D_{10} = \{A_2, A_3\}$,
$D_{11} = \{A_2, A_3, A_4\}$,
$D_{12} = \{A_2, A_4\}$,
$D_{13} = \{A_3\}$,
$D_{14} = \{A_3\}$,
$D_{15} = \{A_4\}$.

3.3.4 Сокращение перебора при построении безызбыточных покрытий

Для достижения результата (нахождения минимального и кратчайшего покрытия) достаточно находить только безызбыточные покрытия (теорма 3.2). Однако не существует эффективного алгоритма поиска всех безызбыточных покрытий, поэтому рассмотрим алгоритм, позволяющий сократить их количество.

Алгоритм заключается в следующем: если текущее подмножество является покрытием, то в него не нужно более вводить новые элементы. Согласно теореме 3.1, безызбыточные покрытия – граница вогнутого множества всех покрытий.

Алгоритм граничного перебора по вогнутому множеству.

0. Текущее подмножество $D = \{A_1\}, i = 0$. Переходим к п.4 алгоритма.

1. Находим наибольший номер j элемента в подмножестве D. Рассматриваем следующие ситуации:

a) если $j \neq n$ и D не покрытие, то переходим к п.3 алгоритма;

b) если $j \neq n$ и D покрытие, то переходим к п.2 алгоритма;

c) если $j = n$, то удаляем A_n из D, и если $D = \emptyset$, то переходим к п.5 алгоритма; иначе – снова находим наибольший номер j элемента в подмножестве D.

2. Удаляем элемент A_j из D.

3. $j = j + 1$. Вводим элемент A_j в D.

4. Исследуем, является ли очередное построение D покрытием. Если нет, то переходим к п.1 алгоритма, иначе:

– удаляем из ранее построенных покрытий избыточные, соответственно сокращая i;

– запоминаем D как покрытие $P_i (i = i + 1;\ P_i = D$;

– переходим к п.1 алгоритма.

5. Заканчиваем построение всех безызбыточных покрытий.

Для ***примера*** такая последовательность построения подмножеств D (покрытия выделены красным, безызбыточные покрытия - зеленым):

$D = \{A_1\}, D_1 = \{A_1, A_2\}$,

$D = \{A_1, A_3\}, D_2 = \{A_1, A_3, A_4\}$,

$D = \{A_1, A_4\}, D = \{A_2\}, D = \{A_2, A_3\}, D_3 = \{A_2, A_3, A_4\}$,

$D = \{A_2, A_4\}, D = \{A_3\}, D_2 = \{A_3, A_4\}$,

$D = \{A_4\}$.

Из пяти возможных безызбыточных покрытий получено только два.

Рассмотренный алгоритм граничного перебора достаточно прост и универсален, он применим для любых задач, в которых возможные решения составляют вогнутое множество. Однако сокращение перебора по этому алгоритму недостаточно, так как алгоритм граничного перебора мало учитывает специфику задачи. Используя представление информации таблицей, можно достигнуть большего сокращения процесса решения.

3.3.5 Сокращение таблицы покрытий до циклической

Теоремы о сокращении таблицы покрытий. Объем перебора существенно зависит от размеров таблицы покрытий, поэтому изложенные в данном разделе способы вычеркивания некоторых строк и столбцов таблицы покрытий имеют большое значение для сокращения перебора.

Теорема (ядро покрытия).

Если в столбце таблицы покрытий содержится единственная единица, то строка, содержащая эту единицу, входит во все покрытия. Эта строка называется **ядерной**.

Множество ядерных строк заранее выделяется и запоминается для введения во все покрытия. Ядерные строки из таблицы удаляются и при этом вычеркиваются все покрытые ими столбцы (т.е. вычеркиваются все столбцы, в которых есть единицы в ядерных строках).

Теорема (антиядро покрытия).

Если после удаления ядерных строк и покрытых ими столбцов в таблице покрытий в какой-либо строке не остается единиц, то эта строка не входит не в одно безызбыточное покрытие. Такие строки называются **антиядерными** и вычеркиваются из таблицы без запоминания.

Поглощающий вектор: вектор E поглощает вектор F (E≥F), если для всех компонентов e_i, f_i этих векторов можно одновременно записать, что $e_i \geq f_i$.

Пример:

M=1101, N=0101, S=1001, T=1111

M ≥ N, M ≤ T, вектора M и S, N и S - несравнимы

M – поглощающий вектор для N

M – поглощаемый вектор для T

Теорема (о поглощающих столбцах).

В таблице покрытий могут быть вычеркнуты все поглощающие столбцы (рассматриваемые как векторы) без ущерба для построения всех безызбыточных покрытий.

Теорема (о поглощаемых строках при поиске одного кратчайшего покрытия).

Если при решении задачи покрытия достаточно гарантировать получение хотя бы одного кратчайшего покрытия, то можно удалять все поглощаемые строки.

Теорема (о поглощаемых строках при построении минимальных покрытий).

Если при решении задачи покрытия достаточно гарантировать построение всех (хотя бы одного) минимальных покрытий, то можно вычеркивать поглощаемую строку, если цена ее больше (или равна) цены поглощающей строки.

Пример:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	a_i
A_1		1				1	1				3
A_2		1			1		1				2
A_3			1		1			1			2
A_4	1									1	3
A_5				1				1			3
A_6					1			1			3
A_7	1				1		1		1		2
A_8						1			1	1	1

В столбцах b_3 и b_4 по одной 1 в каждом, поэтому строки A_3 и A_5, содержащие эти 1, являются ядерными. Эти строки запоминаем для введения во все безызбыточные покрытия этой таблицы и после удаления этих строк и всех покрытых ими столбцов, таблица покрытий сократится до следующей (можно таблицу не перечерчивать, а вычеркнуть соответствующие строки и столбцы):

	b_1	b_2	b_6	b_7	b_9	b_{10}	a_i	
A_1		1	1	1			3	
A_2		1		1			2	
A_4	1					1	3	$\{A_3, A_5\}$
A_6							3	
A_7	1			1	1		2	
A_8			1		1	1	1	

В полученной таблице строка $\mathbf{A_6}$ не содержит 1, поэтому она является антиядерной и вычеркивается из таблицы без запоминания.

Также столбец $\boldsymbol{b_7} \geq \boldsymbol{b_2}$, поэтому поглощающий столбец $\boldsymbol{b_7}$ вычеркивается.

Получаем следующую таблицу:

	$\boldsymbol{b_1}$	$\boldsymbol{b_2}$	$\boldsymbol{b_6}$	$\boldsymbol{b_9}$	$\boldsymbol{b_{10}}$	$\boldsymbol{a_i}$	
$\mathbf{A_1}$		1	1			**3**	
$\mathbf{A_2}$		1				**2**	
$\mathbf{A_4}$	1				1	**3**	$\{A_3, A_5\}$
$\mathbf{A_7}$	1			1		**2**	
$\mathbf{A_8}$			1	1	1	**1**	

В полученной таблице строка $\mathbf{A_2}$ поглощается строкой $\mathbf{A_1}$ ($\mathbf{A_2} \leq \mathbf{A_1}$). Поэтому возможно несколько разветвлений процесса решения:

I. Если нужно построить все безызбыточные покрытия, то нельзя более проводить никакие сокращения строк и полученная таблица далее не сокращается.

II. Если нужно найти минимальные покрытия, то согласно теореме 3.7. нужно обратить внимание на цены $\boldsymbol{a_i}$ строк: $\mathbf{A_2} \leq \mathbf{A_1}$ и $\boldsymbol{a_2} < \boldsymbol{a_1}$, поэтому строку $\mathbf{A_2}$ нельзя вычеркивать и таблица также не упрощается.

III. Если ищем одно кратчайшее покрытие, то строку $\mathbf{A_2}$, как поглощаемую, можно вычеркнуть. Тогда строка $\mathbf{A_1}$ становится ядерной, так как в столбце $\boldsymbol{b_2}$ останется единственная единица. И строка $\mathbf{A_1}$ добавляется ко множеству ядерных строк - $\{\mathbf{A_1}, \mathbf{A_3}, \mathbf{A_5}\}$, таблица покрытий упрощается: вычеркивается строка $\mathbf{A_1}$ и покрытые ею столбцы $\boldsymbol{b_2}$ и $\boldsymbol{b_6}$. Получается следующая таблица, которая уже не упрощается:

	$\boldsymbol{b_1}$	$\boldsymbol{b_9}$	$\boldsymbol{b_{10}}$	$\boldsymbol{a_i}$	
A_4	1		1	**3**	$\{A_1, A_3, A_5\}$
A_7	1	1		**2**	
A_8		1	1	**1**	

Резюмируя все вышесказанное, можно утверждать, что, используя теоремы 3.3 – 3.7, можно получить ядро покрытия (запомнив ядерные строки). После чего возможны два исхода процесса решения:

1) Таблица покрытий после упрощения становится пустой (т.е. вычеркнуты все столбцы). В этом случае множество ядерных строк и является требуемым покрытием (решением).

2) Остаток таблицы покрытий более не упрощается посредством применения теорем 3.3 – 3.7. Получается **циклический остаток** таблицы покрытий. Покрытия для циклического остатка можно строить методами перебора покрытий: граничным перебором или разложением по столбцу (который будет рассмотрен позже). Когда к ядерным строкам добавляются строки покрытия циклического остатка, получается покрытие исходной таблицы покрытий.

Алгоритм построения ядра покрытий и циклического остатка для случая **построения одного кратчайшего покрытия:**

0. Считаем исходную таблицу покрытий текущей таблицей покрытий, а множество ядерных строк – пустым.
1. Определяем ядерные строки и запоминаем множество ядерных строк. Сокращаем таблицу покрытий, вычеркивая ядерные строки и столбцы, покрытые ими.
2. Вычеркиваем антиядерные строки.
3. Вычеркиваем поглощающие столбцы.
4. Вычеркиваем поглощаемые строки.
5. Если в результате выполнения пп.1-4 текущая таблица покрытий изменилась, снова выполняем п.1, иначе преобразования заканчиваем.

При сокращении таблицы покрытий до циклической при **построении одного минимального покрытия алгоритм** остается тем-же, за исключением п.4, который формулируется следующим образом:

4. Вычеркиваем поглощаемые строки, веса которых не меньше весов соответствующих поглощающих строк.

3.3.6 Построение покрытий методом разложения по столбцу

Основная идея метода разложения по столбцу: Каждый столбец таблицы покрытий должен быть покрыт и покрывается только за счет строк, содержащих единицу в этом столбце.

Укрупненный алгоритм метода разложения по столбцу.

1. Исходная таблица покрытий упрощается и полученный циклический остаток и ядро покрытия приписываются начальной вершине (корню) дерева решения.
2. Находится неконечная очередная вершина (лист), при которой имеется непустой остаток таблицы покрытий. Если таких вершин нет, то процесс построения дерева решения заканчивается и выполняется п.3, иначе находится столбец с меньшим числом единиц и проводится разложение по этому столбцу. В каждом варианте разложения остаток таблицы сокращается и, если оказывается пустым, то соответствующая вершина объявляется конечной. И снова выполняется п.2 алгоритма.
3. По дереву решений строятся покрытия. Их оценивают и выбирают наилучшие.

Для упрощения процесса решения выбирается очередной столбец с меньшим числом единиц. В выбранном столбце находится несколько единиц и необходимо рассмотреть варианты его покрытия, в каждом из которых выбирается в покрытие одна строка, содержащая единицы в этом столбце. Строки выбираются в порядке уменьшения числа единиц в них. Выбранная в покрытие строка содержит единицы не только в покрываемом столбце, но и в ряде других, поэтому таблица покрытий сокращается: вычеркивается выбранная строка и все покрытые ею столбцы. В каждом очередном варианте не включаются строки, выбранные в покрытие для предыдущих вариантов разложения. Получается свой остаток таблицы покрытий, к которому

применяется алгоритм сокращения таблицы покрытий, выделяются ядерные строки и приписываются к выбранной для данного варианта строке.

Процесс решения фиксируется в виде дерева, в каждой вершине которого записаны множество введенных в решение строк и соответствующий циклический остаток. Начальной вершине (корню) дерева приписаны циклический остаток и множество ядерных строк исходной таблицы покрытий. Из вершины выходит столько ветвей, сколько единиц в столбце, выбранном для разложения. Дерево достраивается разложением очередного циклического остатка, приписанного вершине, в которой разложение еще не проведено и циклический остаток не пуст. Лист, в котором остаток пуст, называется конечным, для него разложение уже не проводится. Конечные листы дерева решения нумеруются римскими цифрами.

Пример: построения дерева решения

	b_1	b_2	b_3	b_4	b_5	a_i
A_1	1		1		1	**3**
A_2	1	1		1		**2**
A_3	1			1	1	**3**
A_4		1	1			**1**

Ядерных и антиядерных строк нет. Столбец b_1 $(b_1 \geq b_4)$является поглощающим и поэтому вычеркивается. Циклический остаток представлен следующей таблицей:

	b_2	b_3	b_4	b_5	a_i
A_1		1		1	**1**
A_2	1		1		**2**
A_3			1	1	**3**
A_4	1	1			**1**

Множество ядерных строк пусто (нет ядра покрытия). Все столбцы содержат по две единицы, поэтому выбираем первый – b_2.

Ветвление ведется по двум строкам $\mathbf{A_2}$ и $\mathbf{A_4}$.

A_2A_1

	b_3	b_5	a_i
A_1	1	1	**1**
A_3		1	**3**
A_4	1		**1**

I

A_4A_3

	b_4	b_5	a_i
A_1		1	**1**
A_3	1	1	**3**

II

В каждом из этих вариантов таблица сокращается (выделяется соответствующая ядерная строка, присоединяемая к множеству строк при соответствующей вершине).

Т.к. остатки таблиц покрытий после этого становятся пустыми, процесс разложения заканчивается.

1. $\{A_1, A_2\}$, $S_1 = 1 + 2 = 3$.
2. $\{A_3, A_4\}$, $S_2 = 3 + 1 = 4$.

Первое покрытие – минимальное. Оба покрытия – кратчайшие.

3.3.7 Приближенное решение задачи о покрытии.

Если при постановке задачи видно, что даже сокращенный перебор приводит к очень трудоемкому процессу решения, то для получения решения приходится отказываться от построения результата с наилучшей стоимостью (максимального или кратчайшего покрытия), достаточно получить результат, удовлетворяющий необходимым условиям:

$$\bigcup_{j=1}^{l} A_{i_j} = B \qquad (3.1)$$

Оптимальности качества результата гарантий уже нет, но целесообразно получить не самый худший результат (хотя бы безызбыточное, лучше близкое к кратчайшему покрытие). При этом в ущерб качеству, необходимо значительно упростить процесс решения.

Покрытие, близкое к кратчайшему, дает следующий простой алгоритм преобразования таблицы покрытий.

Алгоритм метода минимального столбца – максимальной строки.

1. Исходная таблица считается текущей преобразуемой таблицей покрытий, а множество строк покрытия – пустым.

2. В текущей таблице выделяется столбец с наименьшим числом единиц. Среди строк, содержащих единицы в этом столбце, выделяется одна с наибольшим числом единиц. Эта строка включается в покрытие, текущая таблица сокращается вычеркиванием всех столбцов, в которых выбранная строка имеет единицы. Если в таблице есть невычеркнутые столбцы, то снова выполняется п.2, иначе – покрытие построено. Заметим, что при подсчете числа единиц в строке учитываются единицы в невычеркнутых столбцах.

Пример:

	b_1	b_2	b_3	b_4	b_5
A_1	1		1		1
A_2	1	1		1	
A_3	1			1	1
A_4		1	1		

Решение задачи сводится к выбору столбца b_2 и строки A_2, как имеющей три единицы, после чего остаются только столбцы b_3 и b_5.

	b_3	b_5
A_1	1	1
A_3		1
A_4	1	

Очевидно, что из всех строк выбирается A_1, покрывающая оба оставшихся столбца. Покрытие – $\{A_1, A_2\}$.

Пример:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9
A_1	1		1			1	1		
A_2	1	1		1				1	
A_3		1			1		1		
A_4			1					1	1
A_5				1	1			1	1
A_6			1		1	1	1		1
A_7		1		1		1			

Выбираем столбец b_1(две единицы) и строку A_1 (четыре единицы).
Получаем следующую таблицу:

	b_2	b_4	b_5	b_8	b_9
A_2	1	1		1	
A_3	1		1		
A_4				1	1
A_5		1	1	1	1
A_6			1		1
A_7	1	1			

Выбираем столбец b_2(три единицы) и строку A_2 (три единицы).
Получаем следующую таблицу:

	b_5	b_9
A_3	1	
A_4		1
A_5	1	1
A_6	1	1

Таким образом, получено решение $\{A_1, A_2, A_5\}$, которое на одну строку больше кратчайшего покрытия.

Кстати, если бы по первому рассматриваемому столбцу b_1 была выбрана строка A_2, имеющая столько же единиц, сколько и A_1, то было бы получено кратчайшее покрытие

3.3.8 Метод ветвей и границ в задаче о покрытии

Как показано на примерах в предыдущем подразделе метод разложения по столбцу с применением сокращения таблицы покрытий значительно уменьшает объем вычислений по сравнению с граничным перебором. Возможно еще сократить процесс решения, если ввести в этот метод оценку из метода ветвей и границ. Последний является довольно общим методом, применяемым для решения многих задач. Сначала рассмотрим этот метод без ориентации на конкретную задачу.

3.3.8.1 Основные идеи метода ветвей и границ

Основу метода составляют три преобразования:

1) разложение очередной рассматриваемой задачи на подзадачи,
2) вычисление оценок качества решения подзадач,

3) организация и сокращение процесса рассмотрения подзадач.

Рассмотрим каждый из этих пунктов в отдельности.

1) *Разложение задачи*.

Исходная задача или любая полученная из нее подзадача разлагается на несколько более простых подзадач при выполнении двух условий:

– каждая подзадача имеет ту же формулировку и представление, что и исходная задача;

– объединение результатов всех подзадач содержит все искомые результаты решения исходной задачи.

Разложение изображается в виде дерева процесса решения. Исходная задача помещена в корне дерева, подзадачи – на его листьях. Каждая подзадача проще исходной, и к ней можно снова применить разложение, пока этот процесс не приводит к получению результата решения на очередном листе дерева, тогда этот лист является конечным.

2) *Функция оценки и требования к ней*.

Для каждой подзадачи вычисляется значение функции оценки, которая должна удовлетворять следующим двум требованиям:

– Значение оценки должно быть нижней границей значений функции качества результатов рассматриваемой подзадачи, т.е. значение оценки должно быть меньше или равно значению функции качества каждого из этих результатов (это верно для поиска результатов с минимумом функции качества, при поиске результатов с максимальным значением функции качества оценка должна быть верхней границей). Таким образом, если получена оценка для рассматриваемой подзадачи, то гарантируется, что не найдется ни одного результата с качеством решения лучшим, чем значение оценки.

– Вычисление оценки значительно проще, чем построение всех возможных результатов рассматриваемой подзадачи.

3) *Организация и сокращение процесса решения*.

Процесс решения исходной задачи сводится к ее разложению на подзадачи, оценки подзадач, выбору очередной подзадачи для дальнейшего разложения и сокращению дерева разложения за счет удаления тех подзадач, оценки которых хуже, чем значения функции качества уже полученных результатов.

Общая формулировка метода ветвей и границ:

А) Исходная задача вместе с оценкой приписывается корню дерева.

Б) Если есть еще вершины (не конечные листья дерева), в которых решение еще не доведено до результата, то выбирается одна из них с минимальным значением функции оценки и выполняется п.В, иначе выполняется п.Г.

В) Производится разложение подзадачи, приписанной выбранной вершине. Процесс фиксируется построением новых ветвей дерева, каждой из которых приписывается новая, более простая, подзадача и ее оценка. Если в каком-нибудь варианте разложения подзадача упростилась до построения результата (листа), то вместо оценки вычисляется значение функции качества

результата, и тогда в дереве процесса решения удаляются все листья, значения оценок на которых больше (или равны при поиске хотя бы одного лучшего результата), чем полученное значение функции качества. Затем выполняется п.Б.

Г) По дереву процесса решения формируются результаты с лучшими значениями функции качества.

Процедуры разложения и вычисления оценки не должны быть сложными, т.к. в процессе решения они выполняются многократно, но они должны быть достаточно эффективными, так как от этого зависит сложность дерева процесса решения.

Для каждого класса задач необходимо формулировать свои процедуры разложения и оценки, учитывающие особенности задач этого класса. В данном случае рассматриваются формулировки данных процедур при решении задачи покрытия. В разделе «Теория графов» будут рассмотрены соответствующие процедуры для задачи поиска гамильтонова контура минимальной длины.

3.3.8.2 Формулировка метода ветвей и границ для задачи покрытия

Разложение по столбцу таблицы покрытия полностью удовлетворяет всем требованиям процедуры разложения метода ветвей и границ. Действительно, при разложении получаются подтаблицы, имеющие ту же форму и те же требования при решении, что и исходная таблица. В совокупности таблиц разложения содержатся все покрытия исходной таблицы, при использовании алгоритмов сокращения таблицы сохраняются все безызбыточные (минимальные и кратчайшие) или хотя бы одно кратчайшее (или минимальное) в зависимости от применяемого алгоритма удаления поглощаемых строк таблицы покрытий. Процедура разложения таблицы покрытий по столбцу достаточно проста.

Для вычисления функций оценки можно предложить несколько процедур, различающихся по сложности.

Оценка при построении кратчайших покрытий (обозначается буквой O):

I вариант. Оценка O^A:

$O^A =]x[+ l$, где $x = \frac{n}{k}$

$]x[$ – наименьшее целое число, большее x;

n – число столбцов в таблице покрытий;

k – наибольшее число единиц в строке таблицы покрытий

l – число строк, уже введенных в покрытие на пути к данному листу.

Оценка достаточна груба, но вычисляется очень просто.

II вариант.

Оценка $O^Б = O_д + l$ вычисляется при выполнении следующего неравенства:

$\sum_{i=1}^{O_д} k_i \geq n$, где k_i – число единиц в очередной строке с наибольшим числом единиц, оставшейся после удаления предыдущих выбранных строк.

Пример:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	a_i
A_1	1		1			1	1			1
A_2	1	1		1				1		2,5
A_3		1			1		1			2
A_4			1					1	1	1,5
A_5				1	1			1	1	3
A_6			1		1	1	1		1	7
A_7		1		1		1				1

1) O^A вычисляется:

$$O^A = \left]\frac{n}{k}\right[+ l = \left]\frac{9}{5}\right[+ 0 = 2.$$

2) $O^Б$ вычисляется по следующей процедуре:

сначала выбирается строка A_6 с k_1=5, затем любая из строк с k_2=4 и так как 5+4≥9, то $O^Б$=2+0=2.

Пример:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	a_i
A_1	1		1				1			1
A_2		1			1			1		3
A_3		1	1				1			2
A_4				1				1	1	3
A_5				1		1				2
A_6	1		1		1	1			1	5

1) O^A вычисляется:

$$O^A = \left]\frac{n}{k}\right[+ l = \left]\frac{9}{5}\right[+ 0 = 2.$$

2) $O^Б$ вычисляется по следующей процедуре:

сначала выбирается строка A_6 с k_1=5, затем любые две строки с k_2=3 и k_3=3

и так как 5+3+3≥9, то $O^Б$=3+0=3.

Очевидно, что $O^Б$ – более точная оценка, ее вычисление не намного сложнее, чем вычисление O^A.

Оценка при построении минимальных покрытий:

Вычисление оценки для этого случая более громоздко, можно использовать оценки O^A или $O^Б$ и на их основе построит оценки O^{AM} или $O^{БМ}$ наименьших весов.

$$O^{\text{БМ}} = \sum_{i=1}^{O_j^{\text{A}}} a_i^{min} + \varphi_l$$

$$O^{\text{БМ}} = \sum_{i=1}^{O_j^{\text{Б}}} a_i^{min} + \varphi_l$$

где

a_i^{min} – очередной наименьший вес оставшихся строк таблицы покрытий, после удаления ранее выбранных;

φ_l – сумма весов всех l строк, вошедших в покрытие.

Пример:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	a_i
A_1	1		1			1	1			1
A_2	1	1		1				1		2,5
A_3		1			1		1			2
A_4			1					1	1	1,5
A_5				1	1			1	1	3
A_6			1		1	1	1		1	7
A_7		1		1		1				1

1) O^A вычисляется:

$$O^A = \left]\frac{n}{k}\right[+ l = \left]\frac{9}{5}\right[+ 0 = 2.$$

2) $O^Б$=2+0=2.

3) $O^{\text{АМ}} = O^{\text{БМ}} = 1 + 1 = 2$;

Пример:

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	a_i
A_1	1		1				1			1
A_2		1			1			1		3
A_3		1	1				1			2
A_4				1				1	1	3
A_5				1		1				2
A_6	1		1		1	1			1	5

1) O^A вычисляется:

$$O^A = \left]\frac{n}{k}\right[+ l = \left]\frac{9}{5}\right[+ 0 = 2.$$

2) $O^Б$=3+0=3;

3) O^{AM} = 1+2=3;

4) $O^{БМ}$ = 1+2+2=5.

Реализация метода ветвей и границ для задачи покрытия.

Данный метод конкретизируется сформулированными процедурами разложения и оценки, в которых используются все приемы сокращения остатков таблицы покрытий для каждого варианта разложения. При построении покрытий по дереву процесса решения используется процедура прохода от конечных вершин к корню, которая была сформулирована в алгоритме разложения таблицы покрытий по столбцу.

Пример: Дерево процесса решения при использовании оценки $O^{БМ}$

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	a_i
A_1	1		1			1	1			1
A_2	1	1		1				1		2,5
A_3		1			1		1			2
A_4			1					1	1	1,5
A_5				1	1			1	1	3
A_6			1		1	1	1		1	7
A_7		1		1		1				1

$O^{Б} = 2 + 0 = 2$

$O^{БМ} = 1 + 1 = 2$

Две ветки по b_1:

$\{A_1\}$ (1)

	b_2	b_4	b_5	b_8	b_9	a_i
A_3	1		1			2
A_4				1	1	1,5
A_5		1	1	1	1	3
A_6			1		1	7
A_7	1	1				1

$O^{Б} = 2 + 1 = 3;$

$O^{БМ} = 1 + 1 + 1{,}5 = 3{,}5;$

b_9 – поглощающий столбец;

A_6– поглощаемая (A_5) строка

Все столбцы имеют 2 единицы – поэтому выбираем первый b_2

$\{A_2\}$ (2,5)

	b_3	b_5	b_6	b_7	b_9	a_i
A_3		1		1		2
A_4	1				1	1,5
A_5		1			1	3
A_6	1	1	1	1	1	7
A_7			1			1

$O^{Б} = 1 + 1 = 2;$

$O^{БМ} = 2{,}5 + 1 = 3{,}5;$

1) b_9, b_5 – поглощающие столбцы;

2) A_5– антиядерная строка

Все столбцы имеют 2 единицы – поэтому выбираем первый b_3

$\{A_3\}$, (2)

	b_4	b_8	a_i
A_4		1	1,5
A_5	1	1	3
A_7	1		1

$O^{Б} = 1 + 2 = 3;$

$O^{БМ} = 1 + 2 + 1 = 4.$

$\{A_7, A_5\}$, (1+3)

	b_5	b_8	a_i
A_4		1	1,5
A_5	1	1	3

III

~~F=1+1+3=5~~

$\{A_6\}$, (7)

IV

~~F=2,5+7=9,5~~

$\{A_4\}$, (1,5)

	b_6	b_7	a_i
A_3		1	1,5
A_7	1	1	3

$O^{Б} = 2 + 2 = 4;$

$O^{БМ} = 2{,}5 + 1{,}5 + 2 + +1 = 7.$

Решать не нужно

$\{A_5\}$, (3)

I

~~F=1+2+3=6~~

$\{A_7, A_4\}$, (1+1,5)

	b_9	a_i
A_4	1	1,5

II

~~F=1+2+1+1,5=5,5~~

Решение: P={A_1, A_5, A_7}, F=5

3.3.9 Задача о совместимых подмножествах

Рассмотрим еще одну задачу, имеющую большое число различных интерпретаций, - построение максимальных подмножеств на выпуклом множестве.

3.3.9.1 Отношение совместимости и совместимые подмножества

Задано множество $X = \{x_1, \dots, x_n\}$ и бинарное отношение R несовместимости на множестве X ($R \sqsubseteq X \times X$): пара $(x_i, x_j) \in R$ означает, что элементы x_i и x_j несовместны, т.е. не могут быть одновременно включены в одно совместимое множество $Y \sqsubseteq X$.

Отношение R – антирефлексивно, т.е. для всех $x_i \in X$ пара $(x_i, x_j) \notin R$.

Отношение R – симметрично, т.е. если пара $(x_i, x_j) \in R$, то и пара $(x_j, x_i) \in R$.

Если исходное отношение R не симметрично, то его нужно дополнить до симметричного.

Отношение R при небольшом *n* удобно задавать матрицей отношений:

$$R_{ij} = \begin{cases} 1, \text{при } (x_i, x_j) \in R \\ 0, \text{при} (x_i, x_j) \notin R \end{cases}$$

Подмножество $Y \sqsubseteq X$ совместимо, если никакая пара элементов $x_i \in Y$ и $x_j \in Y$ не является несовместимой, т.е. $(x_i, x_j) \notin R$.

Теорема Если Y_k – совместимое подмножество, то и любое $Y_l \subseteq Y_k$ тоже совместимо, т.е. множество совместимых подмножеств – выпукло.

Согласно этой теореме достаточно строить максимальные совместимые подмножества.

Пример:

$Y_1 = \{x_1, x_4\}$, $R_{1,4} = 0, R_{1,2} = R_{1,3} = R_{4,5} = 1$

R	x_1	x_2	x_3	x_4	x_5
x_1	-	1	1	0	0
x_2	1	-	0	1	0
x_3	1	0	-	0	0
x_4	0	1	0	-	1
x_5	0	0	0	1	-

Таким образом, Y_1 – максимальное совместимое подмножество;

$Y_2 = \{x_2, x_3, x_5\}$ – тоже максимальное совместимое подмножество.

Задача построения совместимых подмножеств возникает, например, при подборе коллектива космического корабля или набора цветов на одной клумбе. Другие интерпретации встретятся нам при построении устойчивых подмножеств вершин графа в задаче его раскраски или при строении максимальных интервалов в задаче минимизации булевых функций.

3.3.9.2 Алгоритм граничного перебора построения максимальных совместимых подмножеств

Модификация алгоритма граничного перебора построения всех безызбыточных (минимальных) покрытий на вогнутом множестве позволяет получить алгоритм граничного перебора на выпуклом множестве совместимых подмножеств.

0. Текущее подмножество $Y = \{x_1\}, i = 0$.

1. Находим наибольший номер j такой, что $x_j \in Y$. Рассматриваем следующие ситуации:

a. Если $j \neq n$ и $i = 0$, то выполняем п.3

b. Если $j \neq n$ и $i \neq 0$, то выполняем п.2

c. Если $j = n$, то удаляем элемент x_n из $Y (Y = Y \backslash \{x_n\})$, и если $Y \neq \emptyset$, то выполняем п.1, если $Y = \emptyset$, то выполняем п.6.

2. Удаляем элемент x_j из $Y\big(Y = Y \backslash \{x_j\}\big)$.

3. $j = j + 1$ и если $j > n$, то выполняем п.4, иначе проверяем совместимость элемента x_j со всеми элементами множества Y. Если хотя бы для одного элемента $x_l \in Y\big(x_j, x_l\big) \in R$, то выполняем п.3, иначе, если для всех $x_l \in Y\big(x_j, x_l\big) \notin R$, то вводим элемент x_j в Y, выполняем п.3.

4. Если $i = 0$, то выполняем п.5, иначе проверяем на поглощение множество Y с множествами $Y_s,\ 1 \leq s \leq i$.

5. $i = i + 1, Y_i = Y$, выполняется п.1. Если да, то выполняем п.1, иначе п.5

6. Простроение всех максимальных совместимых подмножеств $Y_s, 1 \leq s \leq i$ закончено.

Наиболее наглядно алгоритм граничного перебора выполняется в векторном представлении подмножеств Y:

$$\vec{Y} = Y^1 \ldots Y^n, \text{где } y^k = \begin{cases} 1, \text{при } x_k \in Y, \\ 0, \text{при } x_k \notin Y. \end{cases}$$

Пример:

R	x_1	x_2	x_3	x_4	x_5
x_1	-	1	1	0	0
x_2	1	-	0	1	0
x_3	1	0	-	0	0
x_4	0	1	0	-	1
x_5	0	0	0	1	-

x_1	x_2	x_3	x_4	x_5	
1			1		y_1
1				1	y_2
	1	1		1	y_3
	1			1	- $y \subseteq y_3$
		1	1		y_4
		1		1	- $y \subseteq y_3$
			1		- $y \subseteq y_1$
				1	- $y \subseteq y_2$

Таким образом, построены четыре максимальных совместимых подмножества:

$y_1 = \{x_1, x_4\}, y_2 = \{x_1, x_5\}, y_3 = \{x_2, x_3, x_5\}, y_4 = \{x_3, x_4\}$

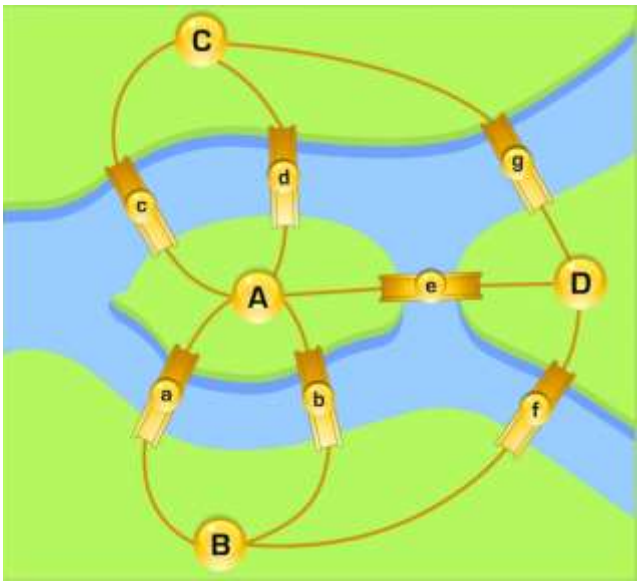
Других максимальных подмножеств не может быть, т.к. при

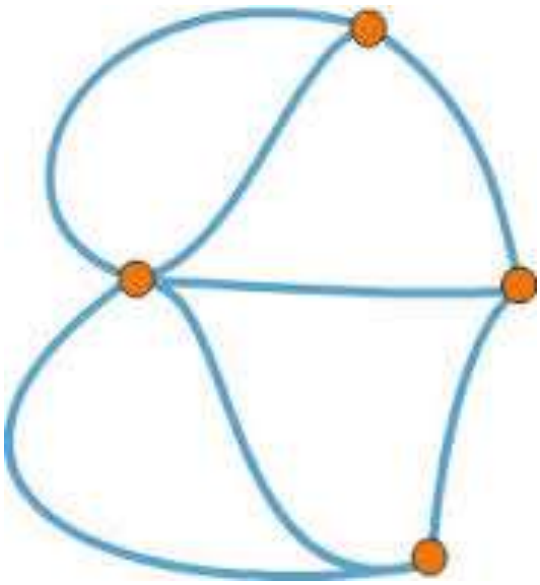
организации решения перебор ведется «плотно», без пропусков подмножеств. Хотя при переборе получаются немаксимальные подмножества, но они сразу же обнаруживаются (y$\subseteq y_s$, $1 \le s \le i$) и не запоминаются.

3.4 ТЕОРИЯ ГРАФОВ

Родоначальником теории графов принято считать математика Леонарда Эйлера (1707-1783).

Историю возникновения этой теории можно проследить по переписке великого ученого. Вот перевод латинского текста, который взят из письма Эйлера к итальянскому математику и инженеру Маринони, отправленного из Петербурга 13 марта 1736 года: «Некогда мне была предложена задача об острове, расположенном в городе Кенигсберге и окруженном рекой, через которую перекинуто семь мостов. Спрашивается, может ли кто-нибудь непрерывно обойти их, проходя только однажды через каждый мост. И тут же мне было сообщено, что никто еще до сих пор не мог это проделать, но никто и не доказал, что это невозможно… После долгих размышлений я нашел легкое правило, основанное на вполне убедительном доказательстве, с помощью которого можно во всех задачах такого рода тотчас же определить, может ли быть совершен такой обход через какое угодно число и как угодно расположенных мостов или не может».





3.4.1 Основные характеристики графов

3.4.1.1 Виды и способы задания графов

Определение

Графом G (V, E) называется совокупность двух множеств – непустого множества V (множества вершин) и множества E его двухэлементных подмножеств множества V (E – множество ребер).

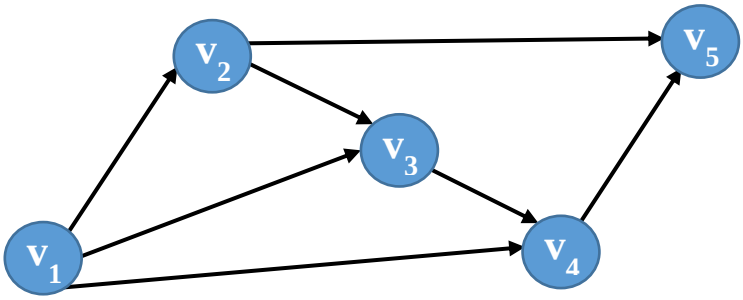
Например:

Для множества $V=\{v_1,v_2,v_3,v_4,v_5\}$ запишем множество всех его двухэлементных подмножеств $V^{(2)}$.

$$V^{(2)} = \{\{v_1,v_2\}, \{v_1,v_3\}, \{v_1,v_4\}, \{v_1,v_5\}, \{v_2,v_3\}, \{v_2,v_4\}, \{v_2,v_5\}, \{v_3,v_4\}, \{v_3,v_5\}, \{v_4,v_5\}\}.$$

Возьмем произвольно некоторое $E \subseteq V^{(2)}$, например:

$$E = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}, \{v_2, v_3\}, \{v_2, v_5\}, \{v_3, v_4\}, \{v_4, v_5\}\}.$$



Для многих задач несущественно, являются ли ребра отрезками прямых или криволинейными дугами; важно лишь то, какие вершины соединяет каждое ребро.

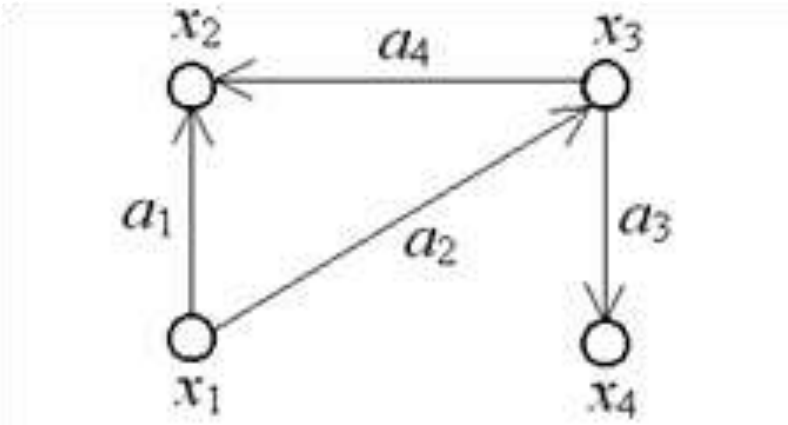
Существуют два основных вида графов (и множество их подвидов): ориентированные и неориентированные.

Если ребрам графа приданы направления от одной вершины к другой, то такой граф называется **ориентированным**. Ребра ориентированного графа называются **дугами**. Соответствующие вершины ориентированного графа называют **началом** и **концом**.

Пример: ориентированный граф $G = (X, A)$.

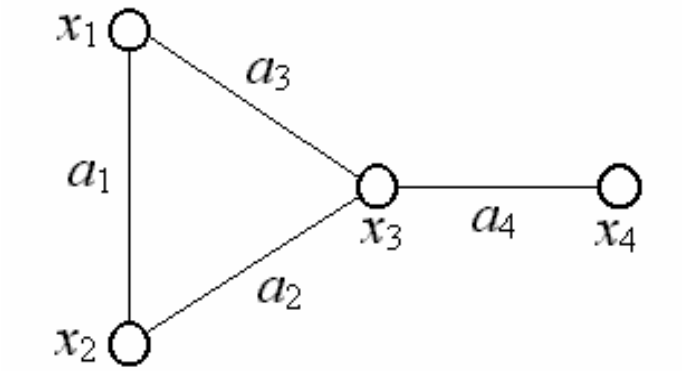
$X = \{x_1, x_2, x_3, x_4\}$,

$A = \{a_1 = (x_1, x_2), a_2 = (x_1, x_3), a_3 = (x_3, x_4), a_4 = (x_3, x_2)\}$.



Если направления ребер не указываются, то граф называется **неориентированным** (или просто графом).

Пример: неориентированный граф G = (X, A).
$X = \{x_1, x_2, x_3, x_4\}$,
$A = \{a_1 = (x_1, x_2), a_2 = (x_2, x_3), a_3 = (x_1, x_3), a_4 = (x_3, x_4)\}$.



Граф называется **простым**, если каждую пару вершин соединяет не более чем одно ребро.

Граф, имеющий как ориентированные, так и неориентированные ребра, называется **смешанным**.

Различные ребра могут соединять одну и ту же пару вершин. Такие ребра называют **кратными**.

Граф, содержащий кратные ребра, называется **мультиграфом**.

Неориентированное ребро графа эквивалентно двум противоположно направленным дугам, соединяющим те же самые вершины.

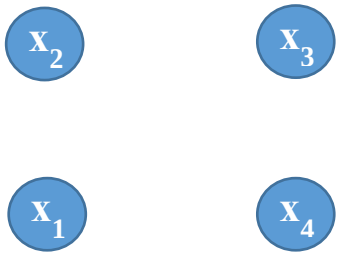
Ребро может соединять вершину саму с собой. Такое ребро называется **петлей**.

Граф с кратными ребрами и петлями называется **псевдографом**.

Множество ребер графа может быть пустым.

Множество вершин графа не может быть пустым.

Пример: ориентированный граф G = (X, A).
X = {x1, x2, x3, x4},
A = ∅ .



Как в случае ориентированного, так и в случае неориентированного ребра говорят, что вершины *x* и *y* **инцидентны** ребру *a*, если эти вершины соединены *a*.

Две вершины называются **смежными**, если они инцидентны одному и тому же ребру.

Два ребра называются **смежными**, если они имеют общую вершину.

Степенью вершины графа называется число ребер, инцидентных этой вершине.

Вершина, имеющая степень 0, называется **изолированной**, а степень 1 – **висячей**.

Необходимость учитывать ориентацию ребер в орграфе приводит к расщеплению понятия «степень вершины» на две части: **полустепенью захода** вершины v_i $(d^-(v_i))$ называется число ребер, заходящих в v_i, **полустепенью исхода** v_i $(d^+(v_i))$ – число ребер, выходящих из нее.

Степень вершины обозначают через $deg\ v_i$.

Для орграфа $deg\ v_i = d^+(v_i) + d^-(v_i)$.

Граф, степени всех *k* вершин которого одинаковы, называется **однородным графом степени *k*.**

Дополнением данного графа называется граф, состоящий из всех ребер и их концов, которые необходимо добавить к исходному графу, чтобы получить полный граф.

Для ориентированного графа множество вершин, в которые ведут дуги, исходящие из вершины *x*, обозначают $G(x)$, то есть

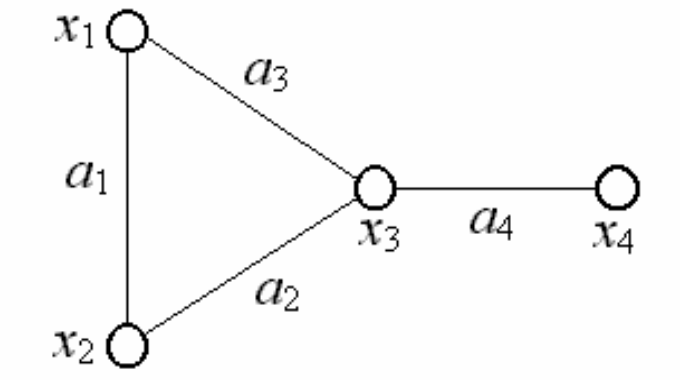
$$G(x) = \{y: (x, y) \in G\}.$$

Множество $G(x)$ называют **образом вершины x**. Соответственно $G^{-1}(y)$ – множество вершин, из которых исходят дуги, ведущие в вершину *y*,

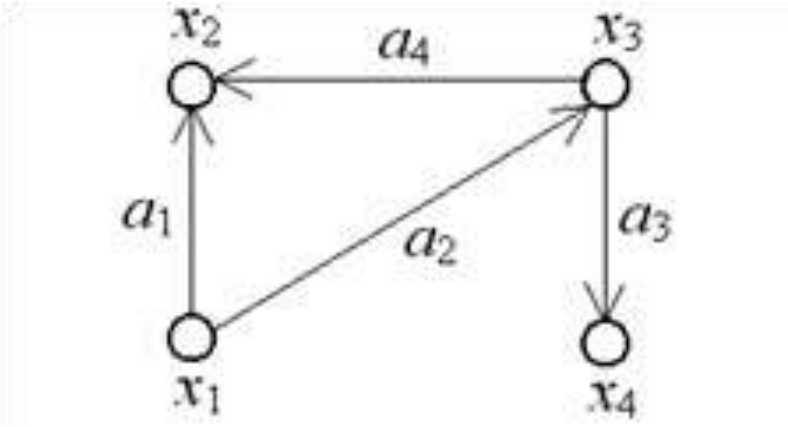
$$G^{-1}(y) = \{x: (x, y) \in G\}.$$

Множество $G^{-1}(y)$ называют **прообразом вершины y.**

Пример:



В графе, изображенном на рис. 2, концами ребра $a1$ являются вершины $x1$, $x2$; вершина $x2$ инцидентна ребрам $a1$, $a2$; степень вершины $x3$ равна 3; вершины $x1$ и $x3$ смежные; ребра $a1$ и $a2$ смежные; вершина $x4$ висячая.

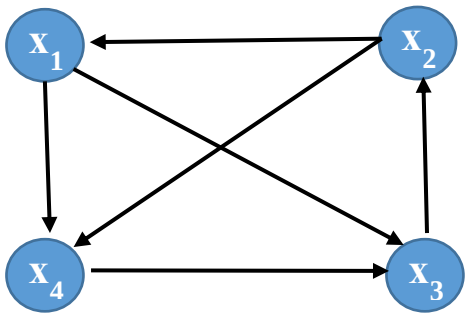


В ориентированном графе, изображенном на рис., началом дуги $a1$ является вершина $x1$, а ее концом – вершина $x2$; вершина x_1 инцидентна дугам a_1 и a_2; $G(x_1) = \{x_2, x_3\}$, $G(x_2) = \varnothing$, $G^{-1}(x_3) = \{x_1\}$, $G^{-1}(x_1) = \varnothing$.

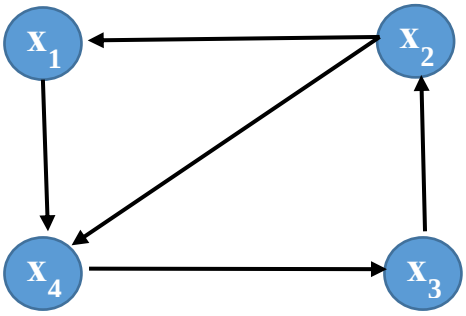
Граф (G_1,X) называется **частичным** для (G,X), если $G_1 \subset G$, т.е. частичный граф получается из графа (G,X) удалением некоторых дуг.

Подграфом (G_1,X_1) графа (G,X) называется граф, все вершины и ребра которого содержатся среди вершин и ребер графа G, т.е. $X_1 \subset X$ и G_1 получается из G удалением дуг, инцидентных удаленным вершинам.

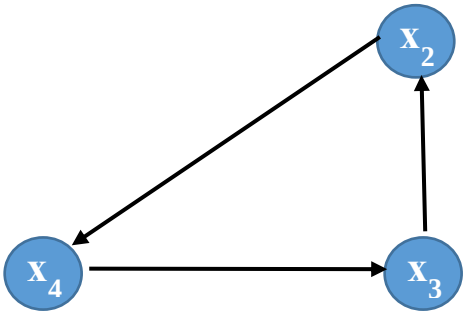
Например: для графа G



Частичный граф:



Подграф:



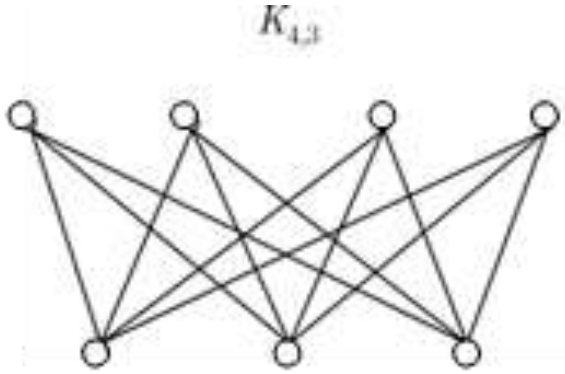
Подграф называется **собственным**, если он отличен от самого графа.

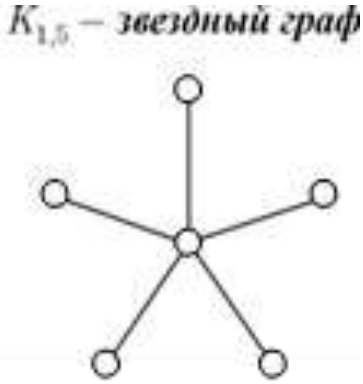
Граф $G = (X, A)$ – **полный**, если для любой пары вершин x_i и x_j существует ребро (x_i, x_j).

Граф $G = (X, A)$ – **симметрический**, если для любой дуги (x_i, x_j) существует противоположно ориентированная дуга (x_j, x_i).

Граф $G = (X, A)$ – **планарный**, если он может быть изображен на плоскости так, что не будет пересекающихся дуг.

Неориентированный граф $G = (X, A)$ – **двудольный**, если множество его вершин X можно разбить на два подмножества X_1 и X_2, что каждое ребро имеет один конец в X_1, а другой в X_2.





Заметим, что граф K_{mn} имеет ровно $m + n$ вершин и $m \cdot n$ ребер.

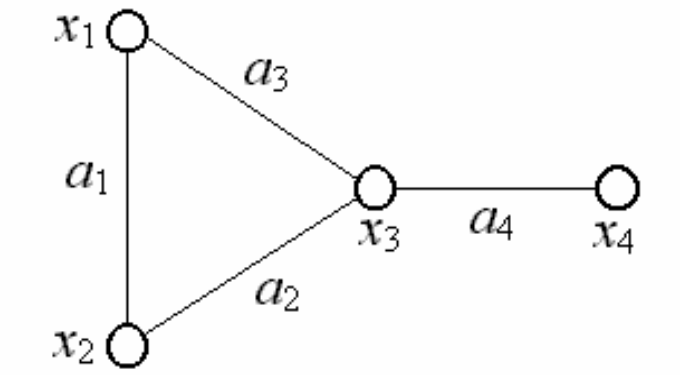
3.4.1.2 Матричные способы задания графов

Для алгебраического задания графов используются матрицы смежности и инцидентности.

Матрица смежности $A=(a_{ij})$ определяется одинаково для ориентированного и неориентированного графов. Это квадратная матрица порядка n, где n – число вершин, у которой

$$a_{ij} = \begin{cases} 1, \text{если } (x_i, x_j) \in A, \\ 0, \text{если } (x_i, x_j) \notin A \end{cases}$$

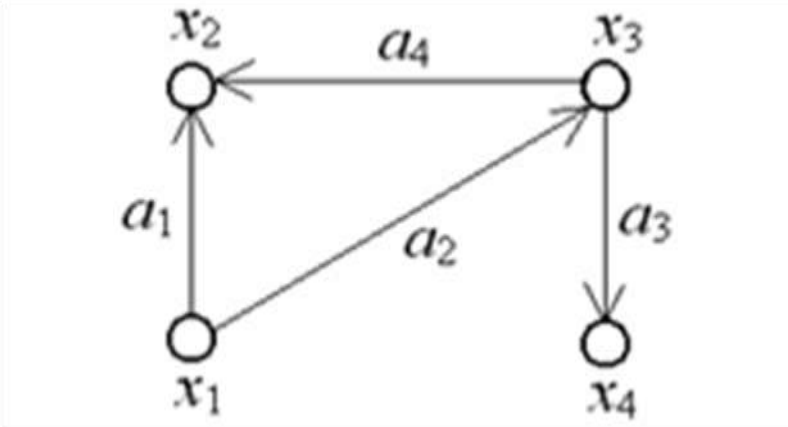
Пример: дан граф



Матрица смежности графа, изображенного на рис., имеет вид:

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

Для ориентированного графа:



Матрица смежности ориентированного графа, изображенного на рис., имеет вид:

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

Матрица смежности полностью задает граф.

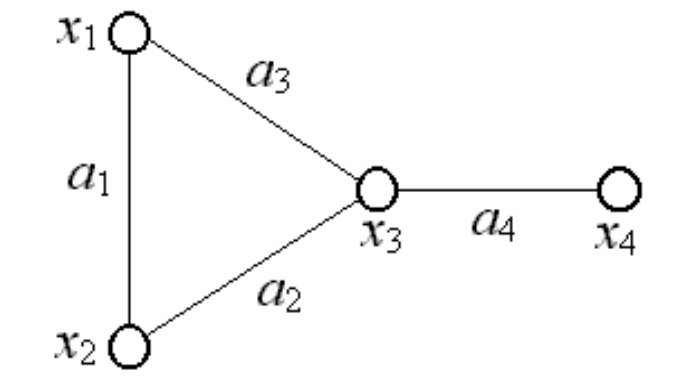
Матрицей инцидентности $B = (b_{ij})$ ориентированного графа называется прямоугольная матрица $(n \times m)$, где n – число вершин, m – число ребер, у которой

$$b_{ij} = \begin{cases} 1, \text{если вершина } x_i \text{является началом дуги } a_j; \\ -1, \text{если вершина } x_i \text{ является концом дуги } a_j; \\ 0, \text{если вершина } x_i \text{ не инцидентна дуге } a_j; \end{cases}$$

Для неориентированного графа матрица инцидентности B задается следующим образом:

$$b_{ij} = \begin{cases} 1, \text{если вершина } x_i \text{инцидентна ребру } a_j; \\ 0, \text{если вершина } x_i \text{ не инцидентна ребру } a_j \end{cases}$$

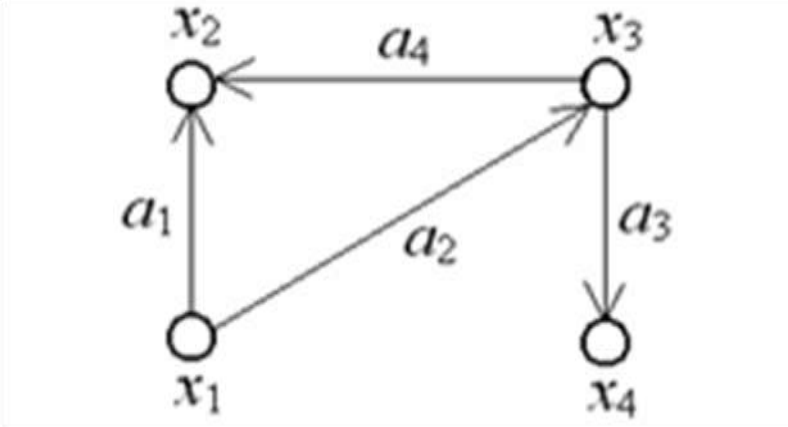
Пример: дан граф



Матрица инцидентности графа:

$$\mathrm{B} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Для ориентированного графа:



Матрица инцидентности ориентированного графа:

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

Матрица инцидентности, так же, как и матрица смежности, полностью задает граф.

Матрицы смежности и инцидентности удобны для задания графов на ЭВМ.

Основные свойства матриц смежности и инцидентности

1. Матрица смежности неориентированного графа является симметричной. Для ориентированного графа это, вообще говоря, неверно.

2. Сумма элементов *i-й* строки или *i-го* столбца матрицы смежности неориентированного графа равна степени вершины x_i.

3. Сумма элементов i-й строки матрицы смежности ориентированного графа равна числу дуг, исходящих из x_i.

4. Сумма элементов *i-го* столбца матрицы смежности ориентированного графа равна числу дуг, входящих в вершину x_i.

5. Сумма строк матрицы инцидентности ориентированного графа является нулевой строкой

Возможны следующие различные способы задания графа:

а) посредством графического изображения;

б) указанием множества вершин и множества ребер (дуг);

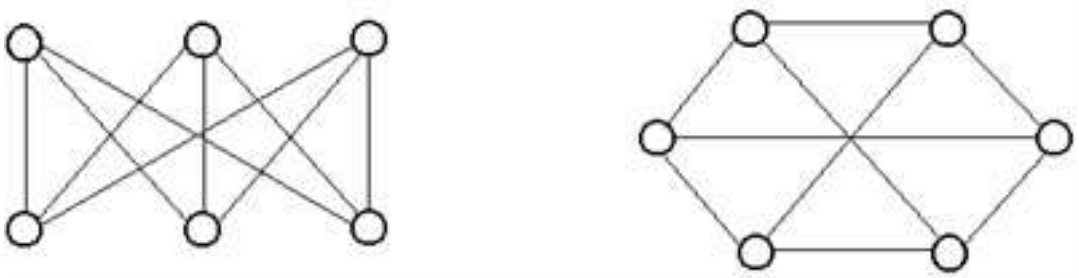
в) матрицей смежности;

г) матрицей инцидентности.

3.4.1.3 Изоморфизм графов

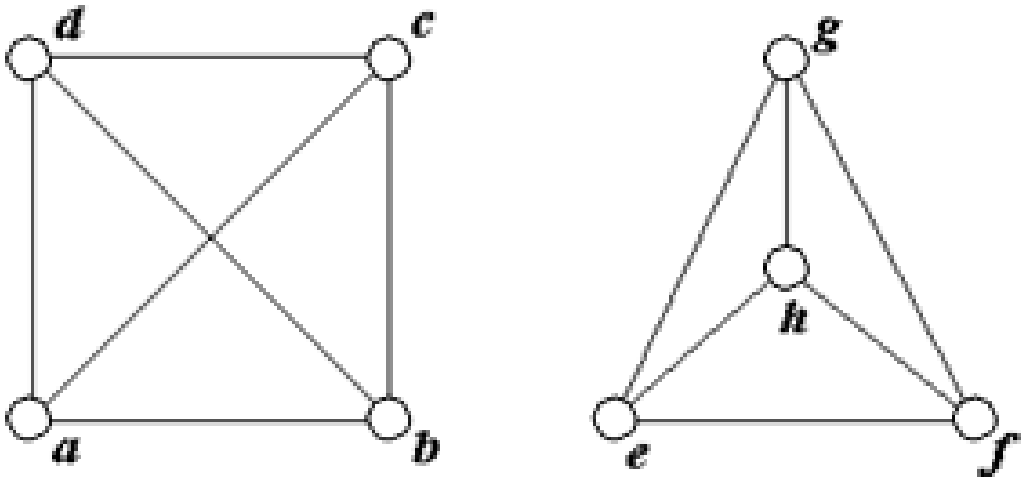
Графы $G1 = (X_1, A_1)$ и $G_2 = (X_2, A_2)$ изоморфны, если существует взаимно однозначное соответствие между множествами вершин X_1 и X_2, такое, что любые две вершины одного графа соединены тогда и только тогда, когда соответствующие вершины соединены в другом графе.

Графы, изображенные на рис. являются изоморфными.



Пример:

Покажем, что следующие два графа изоморфны.



Действительно, на рис. Отображение a → e, b → f, c → g, d → h, являющееся изоморфизмом, легко представить как модификацию первого графа, передвигающую вершину d в центр рисунка.

Изоморфные графы отличаются только нумерацией вершин. Матрицы смежности двух изоморфных графов могут быть получены одна из другой перестановкой строк и столбцов. Чтобы узнать, являются ли два графа

изоморфными, нужно произвести все возможные перестановки строк и столбцов матрицы смежности одного из графов. Если после какой-нибудь перестановки получится матрица смежности второго графа, то эти графы изоморфны. Чтобы убедиться, что графы неизоморфны, надо выполнить все n! возможных перестановок строк и столбцов.

3.4.2 Кратчайший путь

3.4.2.1 Маршруты, циклы в неориентированном графе

Пусть G – неориентированный граф.

Маршрутом или **цепью** в *G* называется такая последовательность (конечная или бесконечная) ребер a_1, a_2,... a_n..., что каждые соседние два ребра a_i и a_{i+1} имеют общую инцидентную вершину.

Одно и то же ребро может встречаться в маршруте несколько раз. В конечном маршруте (a_1, a_2,... a_n) имеется первое ребро a_1 и последнее ребро a_n.

Вершина x_1, инцидентная ребру a_1, но не инцидентная ребру a_2, называется **началом маршрута**, а вершина x_n, инцидентная ребру a_n, но не инцидентная ребру a_{n-1}, называется **концом маршрута**.

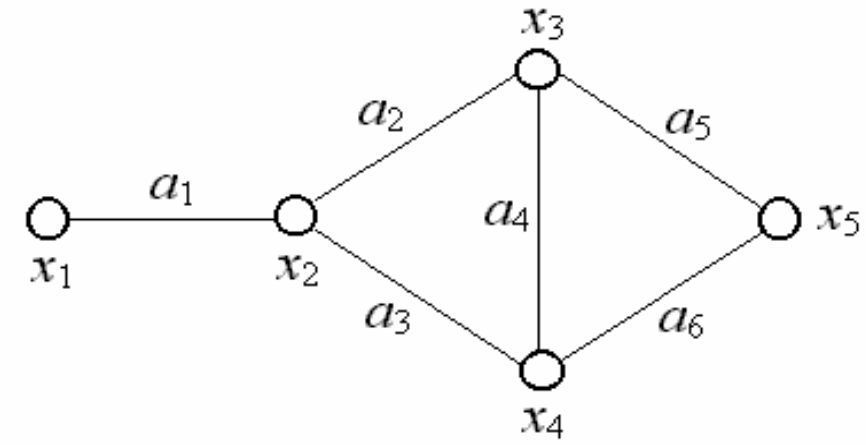
Длиной (или мощностью) маршрута называется число ребер, входящих в маршрут, причем каждое ребро считается столько раз, сколько оно входит в данный маршрут.

Замкнутый маршрут называется **циклом**.

Пример:

В изображенном графе рассмотрим два маршрута из вершины x_1 в вершину x_4:

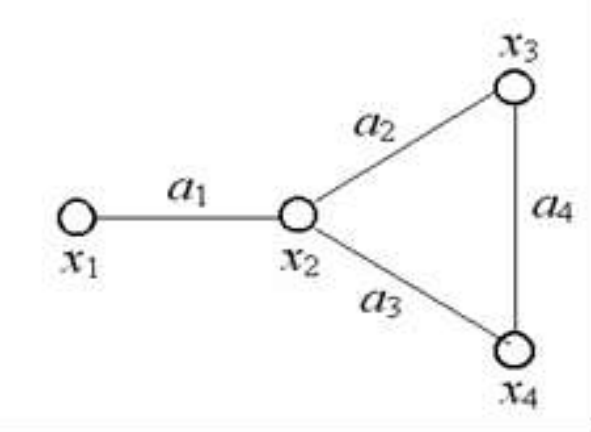
M1 = (a_1, a_2, a_4) и M2 = (a_1, a_2, a_5, a_6).



Длина маршрута M1 равна 3, а длина маршрута M2 равна 4.

Маршрут (цикл), в котором все ребра различны, называется пр**остой цепью (циклом)**. Маршрут (цикл), в котором все вершины, кроме первой и последней, различны, называется **элементарной цепью (циклом)**.

Пример:



В приведенном графе выделим следующие маршруты:

(a1,a3,a4) – простая элементарная цепь длины 3, т.к. все ребра и вершины попарно различны;

(a2,a4,a3) – простой элементарный цикл, т.к. это замкнутый маршрут, у которого все ребра и вершины, кроме первой и последней, различны;

(a1,a2,a4,a3) – цепь, которая является простой, но не элементарной, т.к. все ребра различны, но вершина x2 встречается дважды;

(a1,a2,a2) – маршрут длины 3, не являющийся ни простой, ни элементарной цепью, т.к. ребро a2 и вершина x2 встречаются дважды.

Граф, в котором найдется маршрут, начинающийся и заканчивающийся в одной вершине, и проходящий по всем ребрам графа ровно один раз, называется **эйлеровым графом**.

3.4.2.2 Пути, контуры в ориентированном графе

Понятия пути, контура в ориентированном графе аналогичны понятиям маршрута, цикла в неориентированном графе.

Путем ориентированного графа называется последовательность дуг, в которой конечная вершина всякой дуги, отличной от последней, является начальной вершиной следующей дуги.

Число дуг пути называется **длиной пути**.

Путь называется **контуром**, если его начальная вершина совпадает с конечной вершиной.

Путь (контур), в котором все дуги различны, называется **простым**.

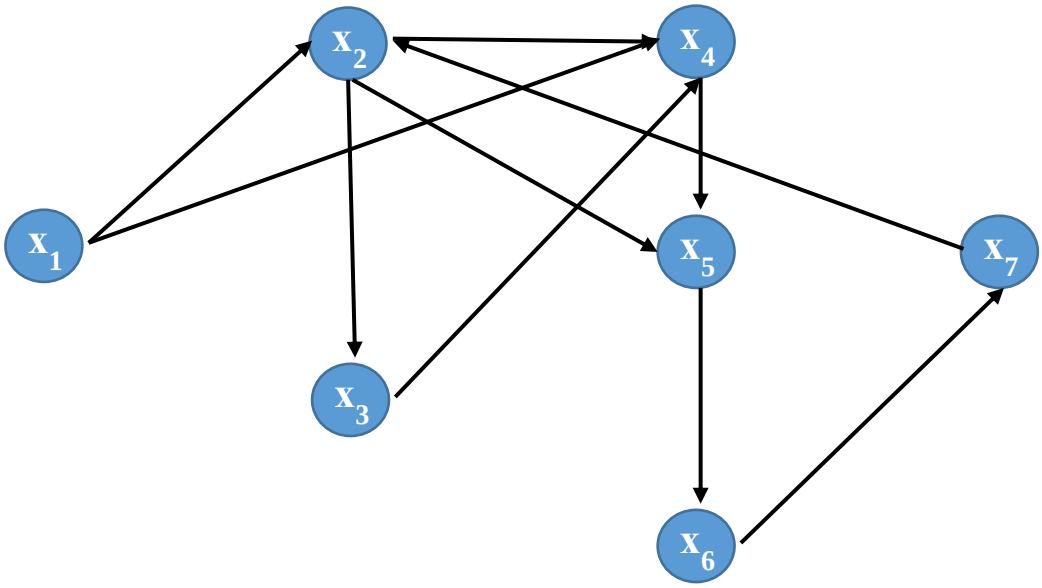
Путь (контур), в котором все вершины, кроме первой и последней, различны, называется **элементарным**.

Следует усвоить, что понятиям ребра, маршрута, цепи, цикла в неориентированном графе соответствуют понятия дуги, пути, ориентированной цепи, контура в ориентированном графе.

Для лучшего запоминания приведем эти термины в таблице.

Неориентированный граф	**Ориентированный граф**
ребро	дуга
маршрут	путь
цикл	контур

Пример



В приведенном графе выделим следующие пути:

(x1,x2,x3,x4) – простой элементарный путь, т.к. каждая вершина и каждая дуга используются не более одного раза;

(x2,x5,x6,x7,x2) – простой элементарный контур, т.к. это замкнутый путь, в котором все дуги и вершины, кроме первой и последней, различны.

3.4.2.3 Пути во взвешенных ориентированных графах

Ориентированный граф называется **нагруженным** (**взвешенным**), если дугам этого графа поставлены в соответствие веса так что дуге (x_i,x_j) сопоставлено некоторое число $l(x_i,x_j)= l_{ij}$, называемое длиной (или весом, или стоимостью) дуги.

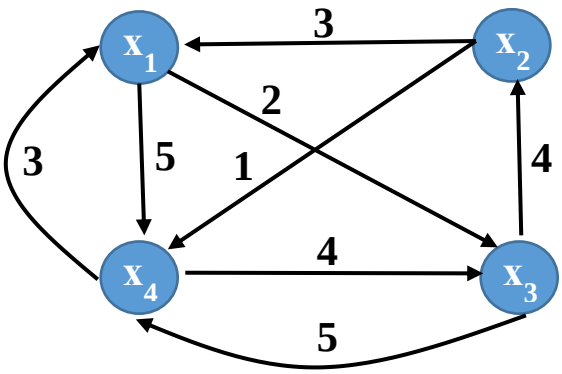
Чаще всего $l_{ij} \geq 0$, во многих приложениях l_{ij} – целочисленна.

Функцию l_{ij} можно задать в матричном виде:

$$l_{ij} = \begin{cases} l_{ij}, \text{при } (x_i, x_j) \in G \\ \infty, \text{при } (x_i, x_j) \notin G \end{cases}$$

Матрица $l= l(x_i,x_j)$ называется **матрицей длин дуг** или **матрицей весов**.

В графическом представлении вес дуги изображается числом на дуге.



l	x_1	x_2	x_3	x_4
x_1	∞	∞	2	5
x_2	3	∞	∞	1
x_3	∞	4	∞	5
x_4	3	∞	4	∞

Длиной (или весом или стоимостью) пути s, состоящего из некоторой последовательности дуг (x_i,x_j), называется число $l(s)$, равное сумме длин дуг, входящих в этот путь, т.е.

$$l(s) = \sum l_{ij},$$

Для невзвешенного (ненагруженного) графа введем понятие **кратчайшего пути**. Это путь с минимальным общим числом дуг, причем каждая дуга считается столько раз, сколько она содержится в этом пути.

3.4.2.4 Волновой алгоритм построения кратчайшего пути в невзвешенном графе

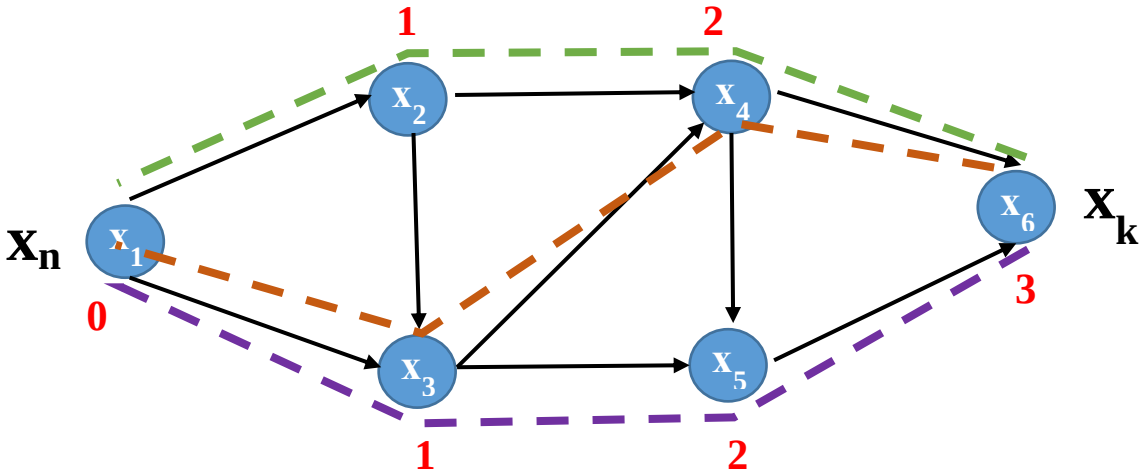
Алгоритм основан на пошаговом распространении волны от вершины начала пути по графу до тех пор, пока волна не коснется вершины конца пути, и затем выделяются дуги, по которым волна дошла до конечной вершины.

0. Вершина x_n помечается нулем, остальные считаются непомеченными; $i=0$.

1. $i=i+1$. Помечаются индексом i все непомеченные ранее вершины, в которых есть дуги от помеченных вершин. Если помечена вершина x_k, то выполняется п.2, иначе, если текущим значением индекса i оказались помеченными какие-либо вершины, то выполняется п.1, иначе делается вывод, что пути из x_n в x_k нет.

2. Обратным проходом по дугам, начиная от вершины x_k, выделяются те дуги и вершины, которые инцидентны выделенным вершинам и разность, между весами которых, равна 1. При движении от вершины x_n по выделенным дугам оказываются построены все кратчайшие пути к вершине x_k.

Пример:



Вершина x_1 помечена 0, вершины x_2 и x_3 помечаются 1, затем вершины x_4 и x_5 помечаются 2 и, наконец, вершина x_6 помечается 3.

Обратный проход: вершина x_6 выделена, разности весов пар вершин x_4x_6 и x_5x_6 равны 1, выделяются дуги (x_4x_6) и (x_5x_6) и вершины x_4 и x_5. От этих выделенных вершин согласно их входам вычисляются разности весов пар вершин x_4x_2, x_4x_3 и x_5x_3, которые равны 1, и выделяются дуги (x_2x_4), (x_3x_4) и (x_3x_5) и вершины x_2 и x_3. Окончательно разность весов пар вершин x_2x_1 и x_3x_1 равна 1, выделяются дуги (x_1x_2) и (x_1x_3). Двигаясь от вершины x_1 по выделенным дугам, получаем три кратчайших пути:

I. (x_1x_2), (x_2x_4), (x_4x_6)

II. (x_1x_3), (x_3x_4), (x_4x_6)

III. (x_1x_3), (x_3x_5), (x_5x_6)

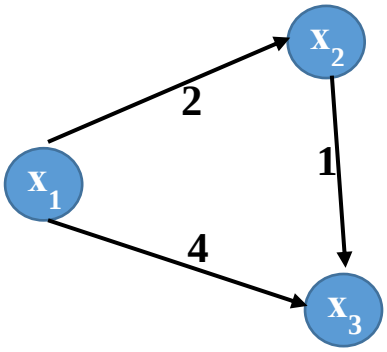
Длины l всех путей одинаковы и равны i шагов распространения волны ($l=i=3$).

3.4.2.5 Волновой алгоритм для взвешенного графа

Особенности построения кратчайшего пути для взвешенного графа волновым алгоритмом.

1. Вместо единичного измерения i (номер шага распространения волны) при расчете пройденного пути для каждой вершины должна использоваться сумма весов l_{ij} дуг (x_i, x_j), ведущих к этой вершине от вершины x_n. К каждой вершине могут идти от вершины x_n несколько путей, суммы длин дуг по разным путям различны. При поиске кратчайшего пути необходимо выбирать меньшую сумму. Волны распространения веса по разным путям доходят до каждой вершины последовательно, при очередной волне необходимо либо оставить старый вес вершины, либо заменить его на новый (меньший). Поэтому при расчете веса вершины x_i за счет волны, подошедшей к ней по дуге (x_j,x_i), вычисление веса V_i определяется по формуле $V_i = \min(V_i, V_j + l_{ij})$.

Пример:



0: $V_1 = 0$
1: $V_2 = 0+2=2$,
$V_3 = 0+4=4$
2: $V_3 = \min(4, 2+1)=3$.

2. Веса вершин в процессе распространения волны могут изменяться неоднократно. При каждом изменении веса вершины V_i это уменьшение веса необходимо передать вершинам исхода x_i, т.е. необходимы специальные средства, отражающие факты получения вершиной нового веса и передачу его другим вершинам. В качестве такого средства используется массив номеров вершин, получивших новый вес (при каждом изменении веса номер вершины включается в этот массив, если его там не было, при передаче веса исключается).

Алгоритм:

1. Вершина x_n получает вес $V_n = 0$, ее номер вводится в массив M номеров вершин, изменивших вес. Остальные вершины x_i получают вес $V_i = \infty$ и их номера не попадают в массив M.

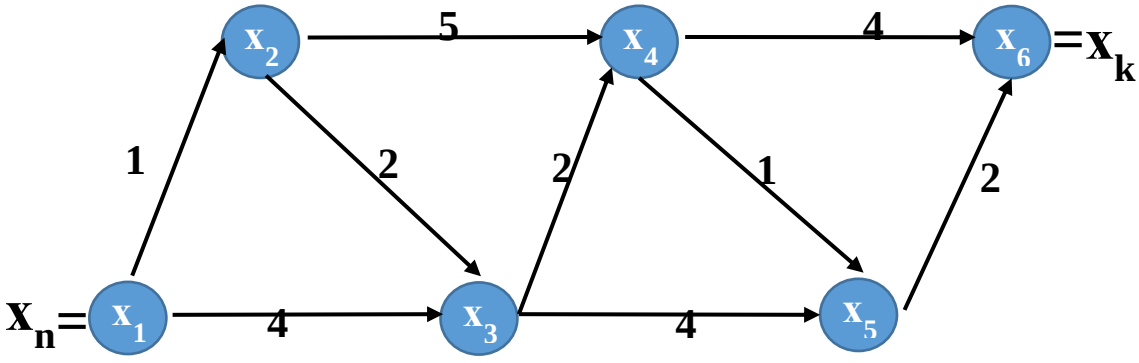
2. Если массив M пуст, то выполняется п.3, иначе выбирается с исключением из него очередная вершина x_i и пересчитываются веса вершин, принадлежащих исходу $G(x_i)$ вершины x_i:

$$\forall x_j \in G(x_i)(V_j = \min(V_j, V_i + l_{ij}))$$

Если вес V_j уменьшается, то номер j включается с приведением подобных в M. Снова выполняется п.2.

3. Если вес $V_k = \infty$, то делается вывод, что пути из вершины x_n к вершине x_k нет, иначе выполняется процедура выделения дуг, такая же, как в волновом алгоритме для невзвешенного графа, за исключением того, что разность весов вершин x_j и x_i должна быть равна l_{ij}. После выделения дуг строятся кратчайшие пути, длины которых равны V_k.

Пример



Решение:

1: $V_n = V_1 = 0, V_2 = V_3 = V_4 = V_5 = V_6 = \infty, M = \{1\}.$

2: $M \neq \emptyset, i = 1, M = \emptyset, G(x_1) = \{x_2, x_3\};$
$V_2 = \min(\infty, 0 + 1) = 1, M = \{2\};$
$V_3 = \min(\infty, 0 + 4) = 4, M = \{2,3\}.$

2: $M \neq \emptyset, i = 2, M = \{3\}, G(x_2) = \{x_3, x_4\};$
$V_3 = \min(4, 1 + 2) = 3, M = \{3\};$
$V_4 = \min(\infty, 1 + 5) = 6, M = \{3,4\}.$

2: $M \neq \emptyset, i = 3, M = \{4\}, G(x_3) = \{x_4, x_5\};$
$V_4 = \min(6, 3 + 2) = 5, M = \{4\};$
$V_5 = \min(\infty, 3 + 4) = 7, M = \{4,5\}.$

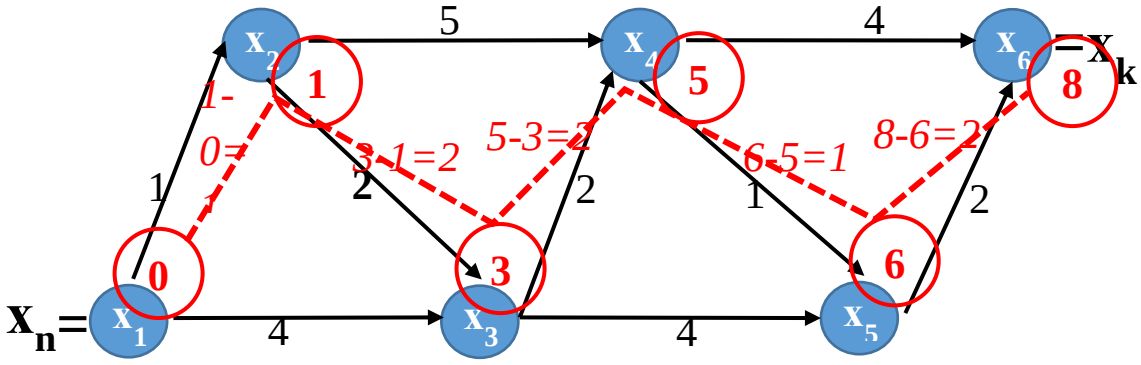
2: $M \neq \emptyset, i = 4, M = \{5\}, G(x_4) = \{x_5, x_6\};$
$V_5 = \min(7, 5 + 1) = 6, M = \{5\};$
$V_6 = \min(\infty, 5 + 4) = 9, M = \{5,6\}.$

2: $M \neq \emptyset, i = 5, M = \{6\}, G(x_5) = \{x_6\};$
$V_6 = \min(9, 6 + 2) = 8, M = \{6\};$

2: $M \neq \emptyset, i = 6, M = \emptyset, G(x_6) = \emptyset;$

2: $M \neq \emptyset$ и выполняется п.3.

Для иллюстрации выполнения п.3 изобразим исходный граф с уже полученными весами вершин, полученными в результате решения.



x_6: $G^{-1}(x_6) = \{x_4, x_5\};$
дуга (x_4, x_6): $(l_{4,6} = 4) \neq (V_6 - V_4 = 3)$ – не выделяется;
дуга (x_5, x_6): $(l_{5,6} = 2) = (V_6 - V_5 = 2)$ – выделяется;
вершина x_5 выделяется.

x_5: $G^{-1}(x_5) = \{x_3, x_4\};$
дуга (x_3, x_5): $(l_{3,5} = 4) \neq (V_5 - V_3 = 3)$ – не выделяется;
дуга (x_4, x_5): $(l_{4,5} = 1) = (V_5 - V_4 = 1)$ – выделяется;
вершина x_4 выделяется.

x_4: $G^{-1}(x_4) = \{x_2, x_3\};$
дуга (x_2, x_4): $(l_{2,4} = 5) \neq (V_4 - V_2 = 4)$ – не выделяется;
дуга (x_3, x_4): $(l_{3,4} = 2) = (V_4 - V_3 = 2)$ – выделяется;
вершина x_3 выделяется.

x_3: $G^{-1}(x_3) = \{x_1, x_2\};$

дуга (x_1, x_3): $(l_{1,3} = 4) \neq (V_3 - V_1 = 3)$ – не выделяется;
дуга (x_2, x_3): $(l_{2,3} = 2) = (V_3 - V_2 = 2)$ – выделяется;
вершина x_2 выделяется.

x_2: $G^{-1}(x_2) = \{x_1\}$;
дуга (x_1, x_2): $(l_{1,2} = 1) = (V_2 - V_1 = 1)$ – выделяется;
вершина x_1=x_k

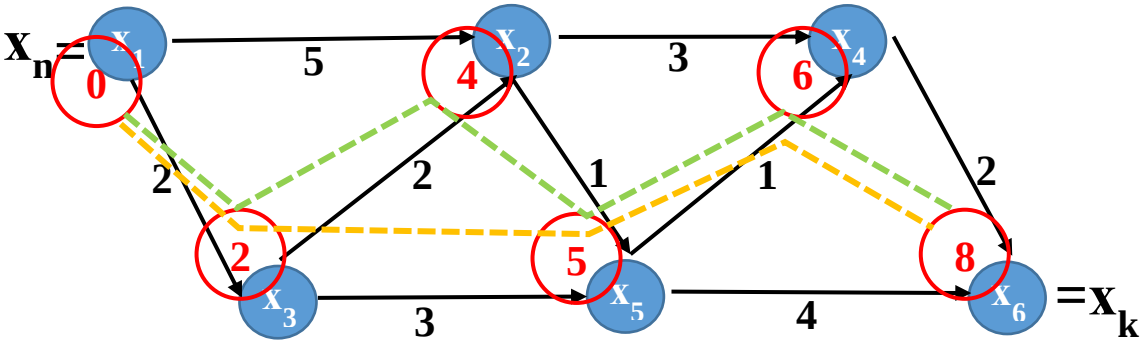
Построен кратчайший путь (x_n, x_k) длиной **8**:

$$(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_5), (x_5, x_6)$$

Процесс решения можно оформить в виде следующей таблицы:

x_1	x_2	x_3	x_4	x_5	x_6	M
0	∞	∞	∞	∞	∞	**1**
0	**1**	4	∞	∞	∞	**2**, 3
0	1	**3**	6	∞	∞	**3**, 4
0	1	3	**5**	7	∞	**4**, 5
0	1	3	5	**6**	9	**5**, 6
0	1	3	5	6	**8**	**6**
0	1	3	5	6	8	Ø

Пример:



x_6: $G^{-1}(x_6) = \{x_4, x_5\}$;
дуга (x_4, x_6): $(l_{4,6} = 4) \neq (V_6 - V_4 = 3)$ – не выделяется;
дуга (x_5, x_6): $(l_{5,6} = 2) = (V_6 - V_5 = 2)$ – выделяется;
вершина x_5 выделяется.

x_5: $G^{-1}(x_5) = \{x_3, x_4\}$;
дуга (x_3, x_5): $(l_{3,5} = 4) \neq (V_5 - V_3 = 3)$ – не выделяется;
дуга (x_4, x_5): $(l_{4,5} = 1) = (V_5 - V_4 = 1)$ – выделяется;
вершина x_4 выделяется.

x_4: $G^{-1}(x_4) = \{x_2, x_3\}$;
дуга (x_2, x_4): $(l_{2,4} = 5) \neq (V_4 - V_2 = 4)$ – не выделяется;
дуга (x_3, x_4): $(l_{3,4} = 2) = (V_4 - V_3 = 2)$ – выделяется;
вершина x_3 выделяется.

x_3: $G^{-1}(x_3) = \{x_1, x_2\}$;
дуга (x_1, x_3): $(l_{1,3} = 4) \neq (V_3 - V_1 = 3)$ – не выделяется;

дуга (x_2, x_3): $(l_{2,3} = 2) = (V_3 - V_2 = 2)$ – выделяется;
вершина x_2 выделяется.

x_2: $G^{-1}(x_2) = \{x_1\}$;
дуга (x_1, x_2): $(l_{1,2} = 1) = (V_2 - V_1 = 1)$ – выделяется;
вершина x_1=x_k

x_1	x_2	x_3	x_4	x_5	x_6	M
0	∞	∞	∞	∞	∞	**1**
0	**5**	2	∞	∞	∞	**2**, 3
0	5	**2**	8	6	∞	**3**, 4, 5
0	4	2	**8**	5	∞	**4**, 5, 2
0	4	2	8	**5**	10	**5**, 2, 5
0	**4**	2	6	5	9	**2**, 6, 4
0	4	2	6	5	**9**	**6**, 4
0	4	2	**6**	5	9	**4**
0	4	2	6	5	**8**	**4**
0	4	2	6	5	8	Ø

Получены два кратчайших пути длиной 8:

1. $(x_1, x_3), (x_3, x_2), (x_2, x_5), (x_5, x_4), (x_4, x_6)$;
2. $(x_1, x_3), (x_3, x_5), (x_5, x_4), (x_4, x_6)$.

3.4.2.6 Алгоритм Дейкстры построения кратчайшего пути во взвешенном графе

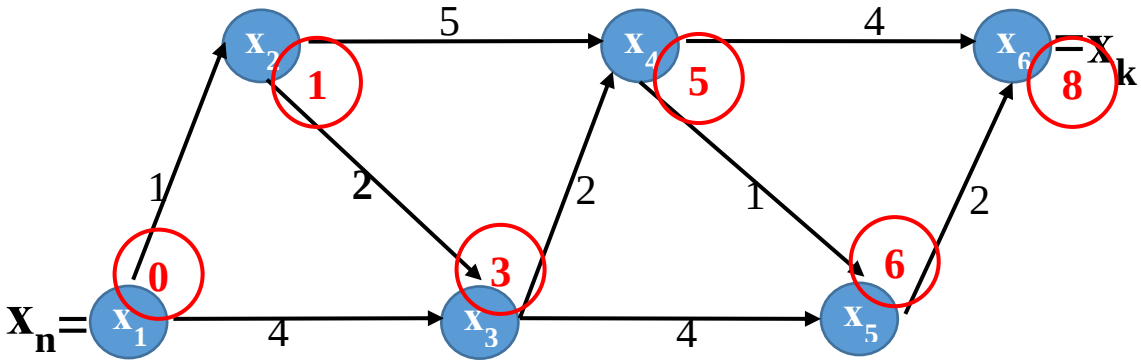
Как видно из последнего примера веса вершин V_i пересчитывались неоднократно и появлялись несколько раз в массиве M, являясь источником новых волн распространения весов. Этот недостаток волнового алгоритма породил развитие других алгоритмов построения кратчайшего пути, одним из которых является алгоритм Дейкстры. Отличия его состоят в том, что на каждом шаге выбирается для распространения веса вершина, имеющая наименьший вес, не распространившая его на другие вершины. Таким образом, в алгоритме Дейкстры нужно отметить вершины уже распространившие свой вес, а из остальных выбирать очередную с наименьшим весом.

Алгоритм:

1. $V_n = 0, \forall x_i (i \neq H)(V_i = \infty)$. Все вершины не отмечены.
2. Среди неотмеченных вершин выбираем x_i с наименьшим весом V_i, вершину отмечаем. Если $V_i = \infty$, то делаем вывод, что пути из вершины x_n в вершину x_k нет, иначе, если выбранная вершина $x_i = x_k$, то выполняем п.3, иначе пересчитываем веса вершин, принадлежащих $G(x_i)$: $V_j = \min(V_j, V_i + l_{ij})$. Выполняем п.2.
3. Аналогичен п.3 волнового алгоритма.

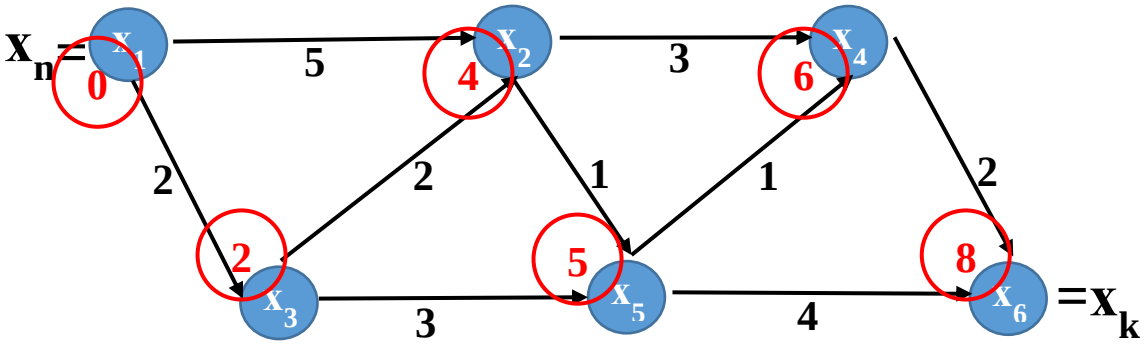
Рассмотрим примеры

Пример:



x_1	x_2	x_3	x_4	x_5	x_6
0	∞	∞	∞	∞	∞
	1	4	∞	∞	∞
		3	6	∞	∞
			5	7	∞
				6	9
					8

Пример:



x_1	x_2	x_3	x_4	x_5	x_6
0	∞	∞	∞	∞	∞
	5	2	∞	∞	∞
	4		∞	5	∞
			7	5	∞
			6		9
					8

Как видно из примеров, процесс решения немного сократился, хотя на каждом шаге необходимо выбирать вершину меньшего веса.

3.4.2.7 Алгоритм построения максимального пути

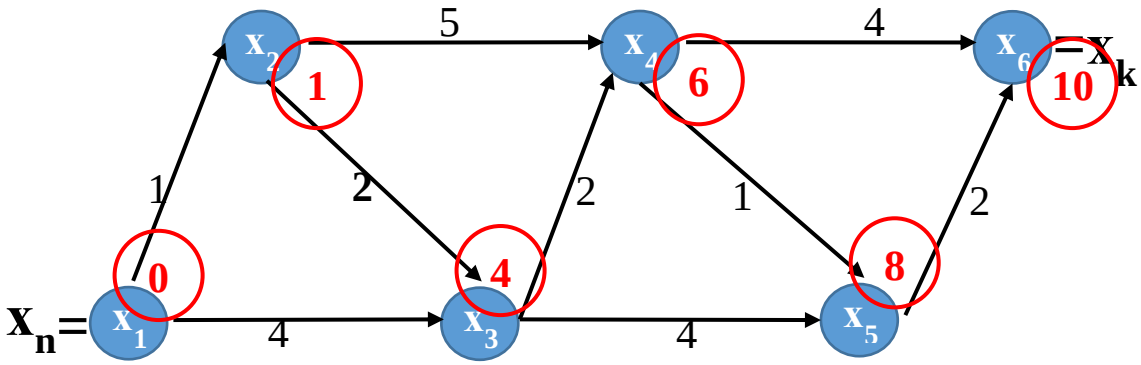
При решении некоторых практических задач возникает необходимость поиска максимального пути (пути с наибольшей суммой длин дуг). Такая задача сводится к задаче нахождения минимального пути заменой процедуры расчета весов вершин

$$V_j = \max(V_j, V_i + l_{ij})$$

и в первом пункте волнового алгоритма присвоением всем вершинам вес 0. Таким образом автоматически будут строиться длиннейшие пути.

При построении длиннейших путей рассматриваются ***элементарные*** или ***простые длиннейшие*** пути, длиннейшие пути ***с заданным выполнением циклов.***

Пример:



x_1	x_2	x_3	x_4	x_5	x_6	M
0	0	0	0	0	0	**1**
0	1	4	0	0	0	**2**, 3
0	1	4	6	0	0	**3**, 4
0	1	4	6	8	0	**4**, 5
0	1	4	6	8	10	**5**, 6
0	1	4	6	8	10	**6**
0	1	4	6	8	10	Ø

Построены три длиннейших пути, длиной 10:

I. $(x_1, x_2), (x_2, x_4), (x_4, x_6)$;

II. $(x_1, x_3), (x_3, x_4), (x_4, x_6)$;

III. $(x_1, x_3), (x_3, x_5), (x_5, x_6)$.

3.4.3 Компоненты связности

3.4.3.1 Определение связности

Неориентированный граф называется **связным**, если каждая пара различных вершин может быть соединена, по крайней мере, одной цепью.

Ориентированный граф называется **сильно связным**, если для любых двух его вершин x_i и x_j существует хотя бы один путь, соединяющий x_i с x_j.

Ориентированный граф называется **односторонне (слабо) связным**, если для любых двух его вершин, по крайней мере, одна достижима из другой.

Компонентой связности неориентированного графа называется его связный подграф, не являющийся собственным подграфом никакого другого связного подграфа данного графа (*максимально связный подграф*).

Компонентой сильной связности ориентированного графа называется его сильно связный подграф, не являющийся собственным подграфом никакого другого сильно связного подграфа данного графа (*максимально сильно связный подграф*).

Компонентой односторонней (слабой) связности неориентированного графа называется его односторонне связный подграф, не являющийся собственным подграфом никакого другого односторонне связного подграфа данного графа (*максимально односторонне (слабо) связный подграф*).

Пусть $G=(X,A)$ неориентированный граф с множеством вершин $X=\{x_1,..,x_n\}$. Квадратная матрица $S = (s_{ij})$ порядка n, у которой

$$s_{ij} = \begin{cases} 1, & \text{если вершины } x_i \text{ и } x_j \text{ принадлежат} \\ & \text{одной компоненте связности} \\ 0, & \text{в противном случае.} \end{cases}$$

называется **матрицей связности** графа G.

Для ориентированного графа квадратная матрица $T = (t_{ij})$ порядка n, у которой

$$t_{ij} = \begin{cases} 1, & \text{если существует путь из } x_i \text{ в } x_j \\ 0, & \text{в противном случае.} \end{cases}$$

называется **матрицей односторонней (слабой) связности (достижимости)**.

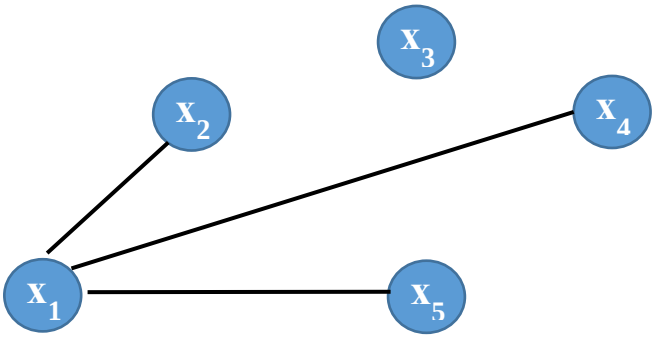
Квадратная матрица $S = (s_{ij})$ порядка n, у которой

$$s_{ij} = \begin{cases} 1, & \text{если существует путь из } x_i \text{ в } x_j \text{ и из } x_j \text{ в } x_i \\ 0, & \text{в противном случае.} \end{cases}$$

называется **матрицей сильной связности**.

Пример:

У неориентированного графа, изображенного на рисунке, две компоненты связности. Первая компонента связности включает вершины x_1, x_2, x_4, x_5, а вторая состоит из одной вершины x_3.



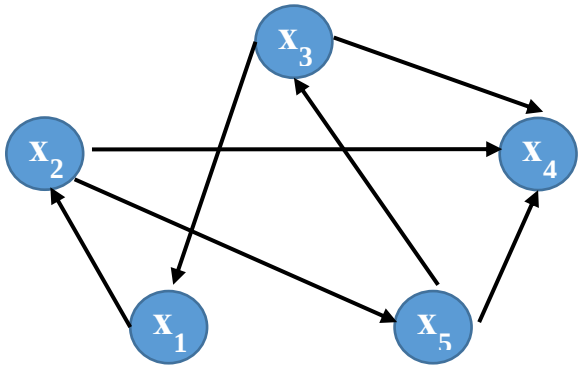
Матрица связности этого графа имеет вид:

$$S=\begin{vmatrix} 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{vmatrix}$$

Мы видим, что 1-ая, 2-ая, 4-ая и 5-ая строки матрицы S одинаковы.

Пример:

У ориентированного графа, изображенного на рисунке, две компоненты сильной связности. Первая компонента связности включает вершины x_1,x_2,x_3,x_5, а вторая состоит из одной вершины x_4. Действительно, для любой пары вершин из множества $\{x_1,x_2,x_3,x_5\}$ существует хотя бы один путь, соединяющий эти вершины. Например, путь (x_1,x_2,x_5,x_3,x_1) соединяет все эти вершины. Из вершины x_4 нет пути ни в одну вершину графа.



Матрица сильной связности этого графа имеет вид:

$$S=\begin{vmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{vmatrix}$$

Мы видим, что 1-ая, 2-ая, 3-ая и 5-ая строки матрицы S одинаковы.

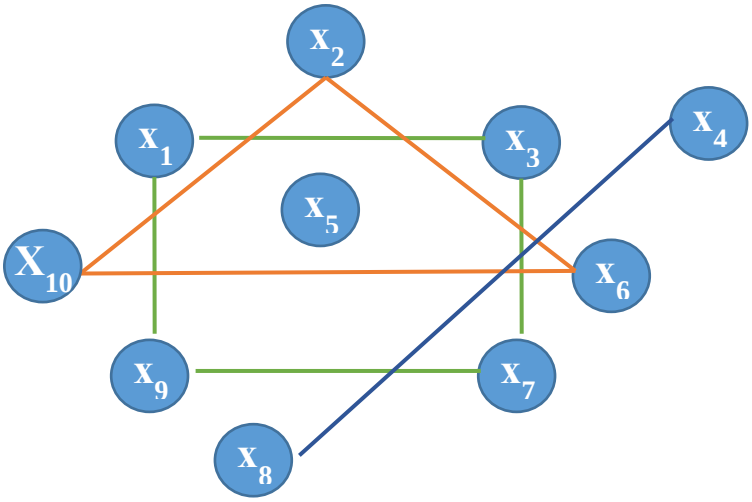
3.4.3.2 Построение компонент связности в неориентированном графе

Как следует из однозначности разбиения неориентированного графа на компоненты связности, их выделение не требует перебора вариантов.

Алгоритм:

1. i=0. Все вершины графа не отмечены.
2. i=i+1. Выбирается очередная неотмеченная вершина, которая отмечается вместе со всеми связанными с нею вершинами значениями индекса i с помощью распространения волны отметок по ребрам, идущим от уже отмеченных индексом i вершин. Таким образом выделяется i-тая компонента связности. Если есть еще неотмеченные вершины, то выполняется п.2, иначе выделение компонент связности закончено.

Пример:



1: i=0

2: i=1. Отмечаем индексом i=1 вершину x_1 и связанные с нею вершины x_3, x_7 и x_9.

Первая компонента связности: $\{x_1,x_3,x_7,x_9\}$

2: i=2. Отмечаем индексом i=2 вершину x_2 и связанные с нею вершины x_6 и x_{10}.

Вторая компонента связности: $\{x_2,x_6,x_{10}\}$

2: i=3. Отмечаем индексом i=3 вершины x_4 и x_8.

Третья компонента связности: $\{x_4,x_8\}$

2: i=4. Отмечаем индексом i=4 вершину x_5.

Четвертая компонента связности: $\{x_5\}$

3.4.3.3 Построение компонент связности в ориентированном графе

Алгоритм заключается к сведению исходного ориентированного графа к неориентированному путем преобразований, не нарушающих сильную связность вершин графа. Для полученного неориентированного графа применяется алгоритм из п.5.3.2.

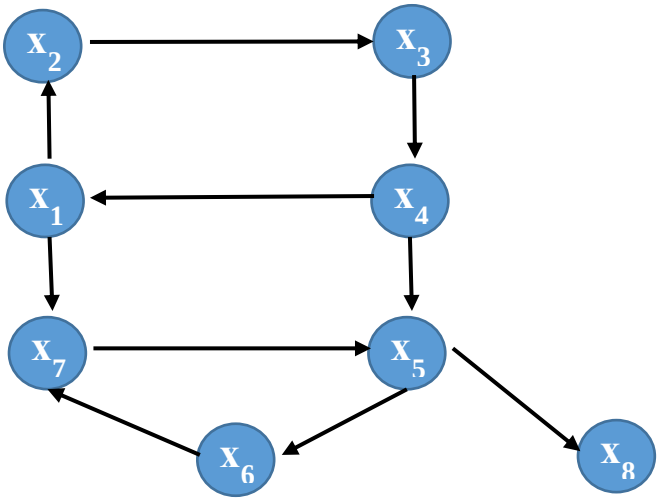
Алгоритм сведения ориентированного графа к неориентированному:

1. Все встречные дуги в исходном орграфе заменяем ребрами.

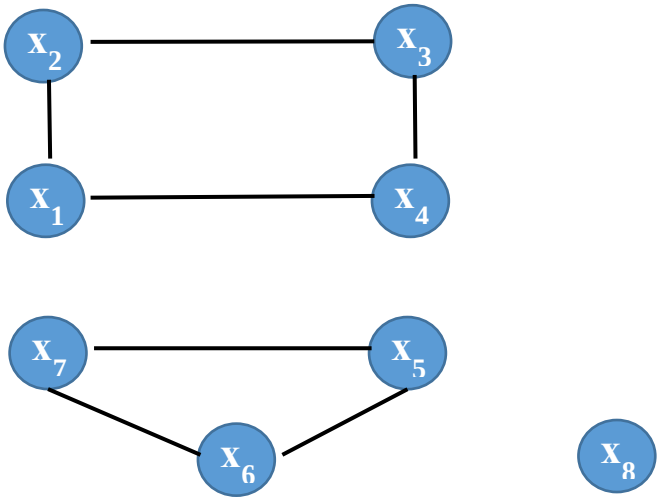
2. Если в преобразуемом графе уже нет дуг, то процедура завершена, иначе выбирается произвольная дуга (x_i,x_j) и, если в графе есть путь (x_j,x_i), то дуга (x_i,x_j) и все дуги пути (x_j,x_i) заменяются ребрами, если же пути (x_j,x_i) в графе нет, то дуга (x_i,x_j) удаляется из графа, т.к. должна сохраняться только сильная связность. Выполняется п.2.

Пример:

Исходный орграф:



Преобразованный неориентированный граф:



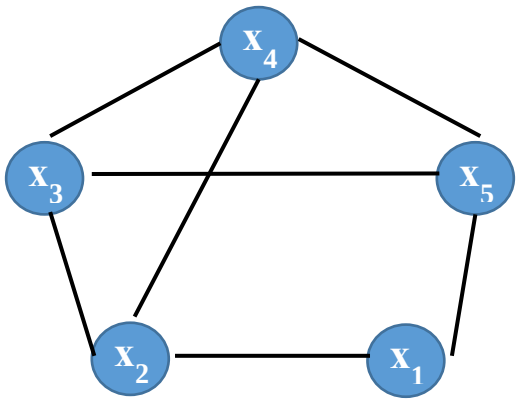
Компоненты связности:

1. $\{x_1,x_2,x_3,x_4\}$
2. $\{x_5,x_6,x_7\}$
3. $\{x_8\}$

3.4.4 Система независимых циклов

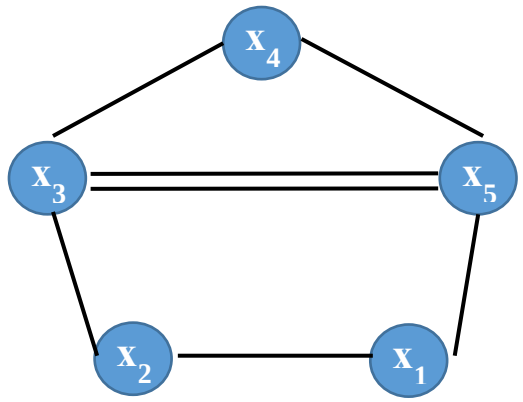
3.4.4.1 Определения

Пример:



Цикл называется **линейно зависимым** от некоторой совокупности других циклов, если его можно построить *линейной комбинацией* циклов этой совокупности.

Чтобы пояснить понятие линейной комбинации, рассмотрим совокупность двух циклов представленного графа:



I. $(x_1, x_2), (x_2, x_3), (x_3, x_5), (x_5, x_1)$;

II. $(x_3, x_5), (x_5, x_4), (x_4, x_3)$.

и организуем их обход. Исключая из обхода ребро (x_3, x_5), пройденное туда и обратно, получаем новый цикл:

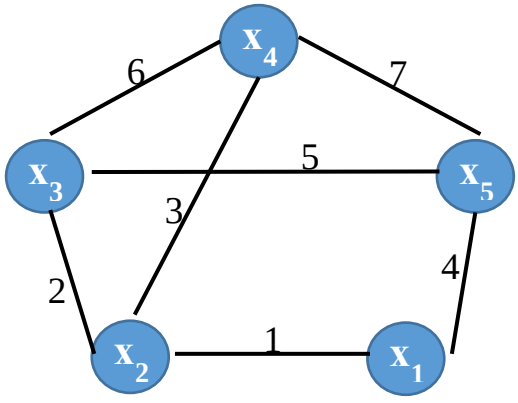
$$(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_5), (x_5, x_1)$$

Этот цикл построен в результате линейной комбинации исходных циклов и является **линейно-зависимым** от них. Таким образом, непосредственно на графе неформально пояснено понятие линейной зависимости.

Формализуем рассмотренное понятие. Сопоставим каждому циклу двоичный m-разрядный вектор, где m – число ребер графа. Пронумеруем ребра. Для i-го цикла компоненты C_{ij} вектора C_i определяется следующим образом:

$$C_{ij} = \begin{cases} 0, \text{если ребро } j \text{ не входит в цикл} \\ 1, \text{если ребро } j \text{ входит в цикл} \end{cases}.$$

Тогда линейной комбинацией векторов C_i называется результат векторной операции сложения по модулю два. Для данного примера:



	1	***2***	***3***	***4***	***5***	***6***	***7***
⊕	1	1	0	1	1	0	0
	0	0	0	0	1	1	1
	1	1	0	1	0	1	1

⊕	C_1
	C_2
	$C_1 \oplus C_2$

Системой независимых циклов графа называется максимальная линейно независимая совокупность циклов графа.

Граф в общем случае может иметь не одну систему независимых циклов, однако число Υ циклов в системе для каждого графа определяется в его структуре однозначно. Число Υ называется **цикломатическим числом графа.**

Цикломатическое число графа указывает то наименьшее число рёбер, которое нужно удалить из данного графа, чтобы добиться отсутствия у графа циклов.

Цикломатическое число всегда неотрицательно.

Теорема 5.1. Мощность Υ системы независимых циклов в графе определяется **формулой Эйлера**:

$$\Upsilon=m-n+p,$$

где m – число ребер;

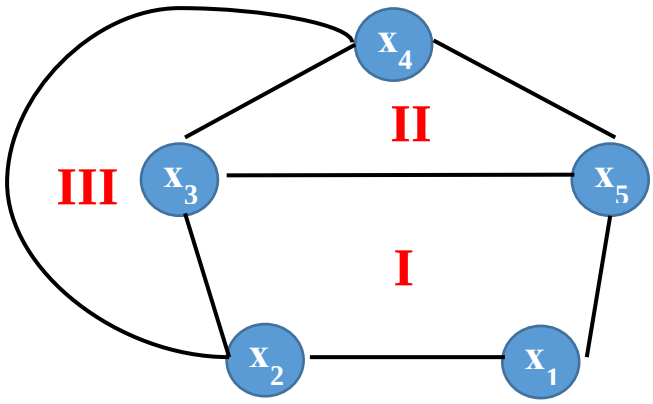
n – число вершин;

p – число компонент связности.

Для рассмотренного примера $\Upsilon=7-5+1=3$, т.е. в систему включаются 3 взаимно независимых цикла, каждый следующий цикл уже можно построить их линейной комбинацией.

Теорема 5.2. Края конечных граней планарного графа составляют систему независимых циклов.

Пример5.6: граф для этого примера – планарный:

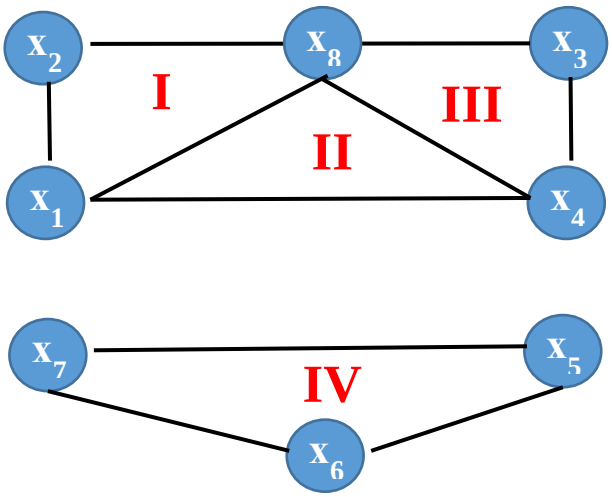


Цикломатическое число:
$\Upsilon=7-5+1=3$

Система независимых циклов этого графа содержит $\Upsilon=3$ цикла:

I. $(x_1, x_2), (x_2, x_3), (x_3, x_5), (x_5, x_1)$;
II. $(x_3, x_5), (x_5, x_4), (x_4, x_3)$;
III. $(x_2, x_3), (x_3, x_4), (x_4, x_2)$.

Пример:



Цикломатическое число:

Υ=10-8+2=4

Система независимых циклов этого графа содержит Υ=4 цикла:

I. $(x_1, x_2), (x_2, x_8), (x_8, x_1)$;

II. $(x_1, x_8), (x_8, x_4), (x_4, x_1)$;

III. $(x_4, x_8), (x_8, x_3), (x_3, x_4)$;

IV. $(x_5, x_6), (x_6, x_7), (x_7, x_5)$

Т.к. большинство графов не являются плоскими, то построение системы независимых циклов является затруднительным. Поэтому обобщенный алгоритм построения системы независимых циклов будет рассмотрен в п. 5.4.3

3.4.4.2 Остов графа

Неориентированным деревом (или просто деревом) называется связный граф без циклов.

Этому определению эквивалентны, как легко показать, следующие определения:

а) дерево есть связный граф, содержащий n вершин и n – 1 ребер;

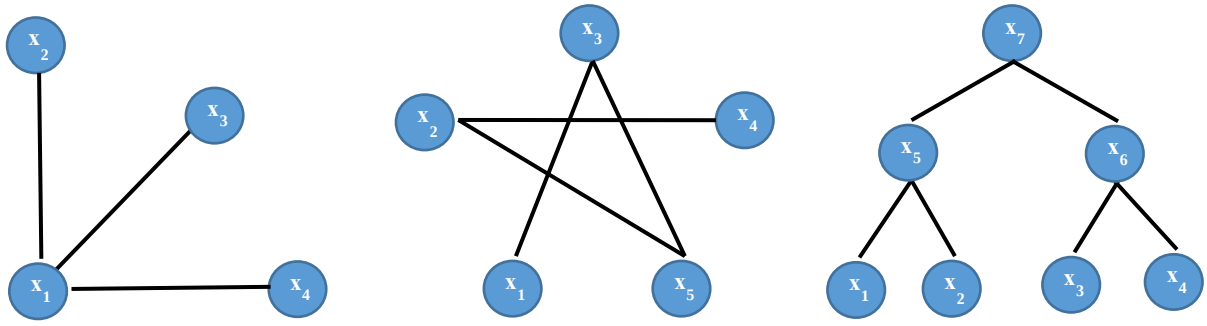
Это следует из свойств отсутствия циклов (Υ=0) и связности (p=1):

Υ=m-n+1,

m=n-1

б) дерево есть граф, любые две вершины которого можно соединить простой цепью.

Примеры:



Если граф несвязный и не имеет циклов, то каждая его связная компонента будет деревом. Такой граф называется **лесом**. Можно интерпретировать предыдущий пример, как лес, состоящий из трех деревьев.

Вершины дерева, которым инцидентно только одно ребро, называются **листьями**.

Одна з вершин может быть объявлена **корнем дерева**.

Теорема. Существует $D_n = n^{n-2}$ различных деревьев на n вершинах. (Доказательство представлено в учебном пособии стр.72-73)

Остовным деревом (или остовом) связного графа G называется любой его частичный граф, содержащий все вершины графа G и являющийся деревом.

В общем случае для графа можно построить несколько остовов.

Пример:

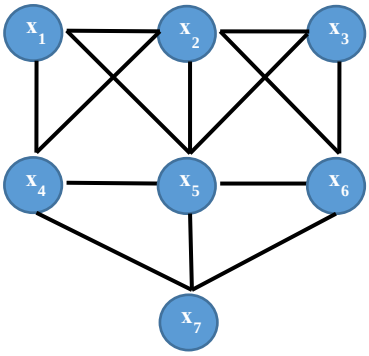


Рис.1а – Граф G

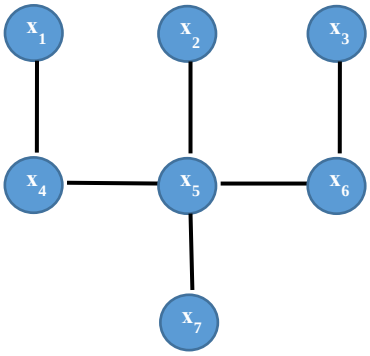


Рис.1б–Остовное дерево

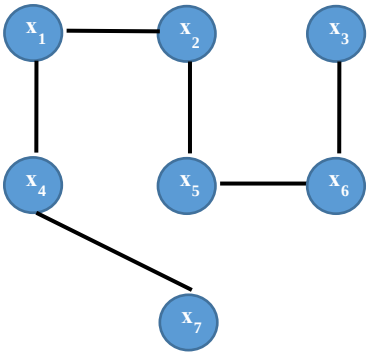


Рис.1в–Остовное дерево

Алгоритм построения произвольного остова:

1. Для каждой компоненты i графа выполняются п.2 и п.3
2. Строим частичный граф, содержащий все n_i вершин компоненты и не содержащий ребер (нулевой граф).
3. Если в текущий частичный граф включены уже $n_i - 1$ ребер, то остов для компоненты i построен, иначе выбираем очередное не рассмотренное ребро компоненты и пытаемся его включить в текущий граф. Если это не приводит к образованию цикла, то ребро включаем, иначе – нет. Выполняем п.3.

Пусть G – связный нагруженный граф. Задача построения минимального остовного дерева заключается в том, чтобы из множества остовных деревьев найти дерево, у которого сумма длин ребер минимальна.

Приведем типичные случаи, когда возникает необходимость построения минимального остовного дерева графа.

а) Нужно соединить n городов железнодорожными линиями (автомобильными дорогами, линиями электропередач, сетью трубопроводов и т.д.) так, чтобы суммарная длина линий или стоимость была бы минимальной.

б) Требуется построить схему электрической сети, в которой клеммы должны быть соединены с помощью проводов наименьшей общей длины.

Задачу построения минимального остовного дерева можно решить с помощью следующего алгоритма.

Алгоритм построения минимального остовного дерева для нагруженного графа (Алгоритм Краскала)

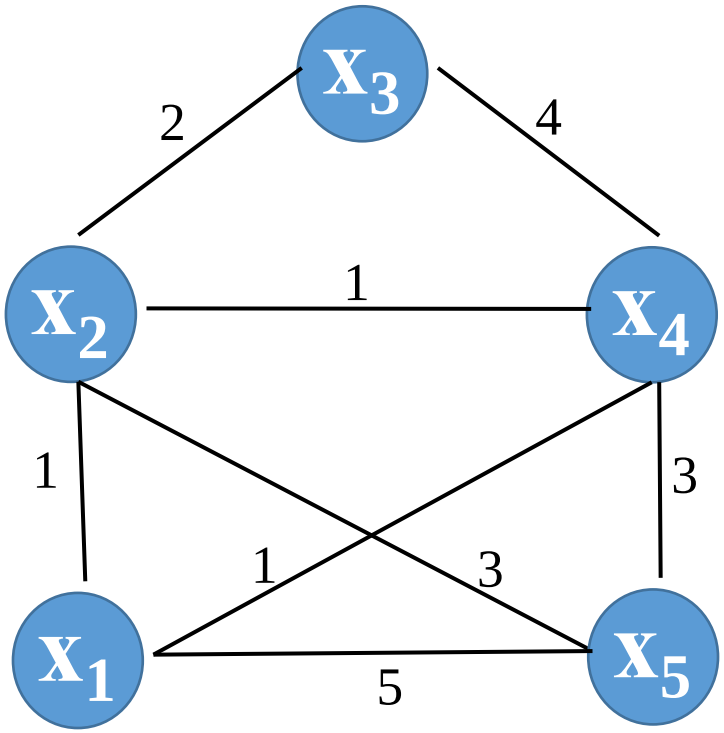
1. Установка начальных значений. Вводится матрица длин ребер C графа G.

2. Выбрать в графе G ребро минимальной длины. Построить граф G_2, состоящий из данного ребра и инцидентных ему вершин. Положить $i = 2$.

3. Если $i = n$, где n – число ребер графа, то закончить работу (задача решена), в противном случае перейти к п. 4.

4. Построить граф $G_i + 1$, добавляя к графу G_i новое ребро минимальной длины, выбранное среди всех ребер графа G, каждое из которых инцидентно какой-нибудь вершине графа G_i и одновременно инцидентно какой-нибудь вершине графа G, не содержащейся в G_i. Вместе с этим ребром включаем в $G_i + 1$ и инцидентную ему вершину, не содержащуюся в G_i. Присваиваем $i := i + 1$ и переходим к п. 3.

Пример:. Найдем минимальное остовное дерево для графа:



Шаг 1. Установка начальных значений. Введем матрицу длин ребер C:

$$C=\begin{vmatrix} \infty & 1 & \infty & 1 & 5 \\ 1 & \infty & 2 & 1 & 3 \\ \infty & 2 & \infty & 4 & \infty \\ 1 & 1 & 4 & \infty & 3 \\ 5 & 3 & \infty & 3 & \infty \end{vmatrix}$$

Шаг 2. Выберем ребро минимальной длины. Минимальная длина ребра равна единице. Таких ребер три: (x_1, x_2), (x_1, x_4), (x_2, x_4).

В этом случае можно взять любое. Возьмем (x_1, x_2).

Построим граф $G2$, состоящий из данного ребра и инцидентных ему вершин $x1$ и $x2$. Положим $i = 2$.

Шаг 3. Так как $n = 5$, то $i \neq n$, поэтому переходим к шагу 4.

Шаг 4. Строим граф $G3$, добавляя к графу $G2$ новое ребро минимальной длины, выбранное среди всех ребер графа G, каждое из которых инцидентно одной из вершин $x1$, $x2$ и одновременно инцидентно какой–нибудь вершине графа G, не содержащейся в $G2$, т.е. одной из вершин $x3$, $x4$, $x5$. Таким образом, нужно выбрать ребро минимальной длины из ребер $(x1, x4)$, $(x1, x5)$, $(x2, x3)$, $(x2, x4)$, $(x2, x5)$. Таких ребер длины единица два: $(x1, x4)$ и $(x2, x4)$.

Можно выбрать любое. Возьмем $(x1, x4)$. Вместе с этим ребром включаем в $G3$ вершину $x4$, не содержащуюся в $G2$. Полагаем $i = 3$ и переходим к шагу 3.

Шаг 3. Так как $i \neq n$, поэтому переходим к шагу 4. *Шаг 4*. Строим граф $G4$, добавляя к графу $G3$ новое ребро минимальной длины из ребер $(x1, x5)$, $(x2, x3)$, $(x2, x5)$, $(x4, x5)$. Такое ребро длины два одно: $(x2, x3)$.

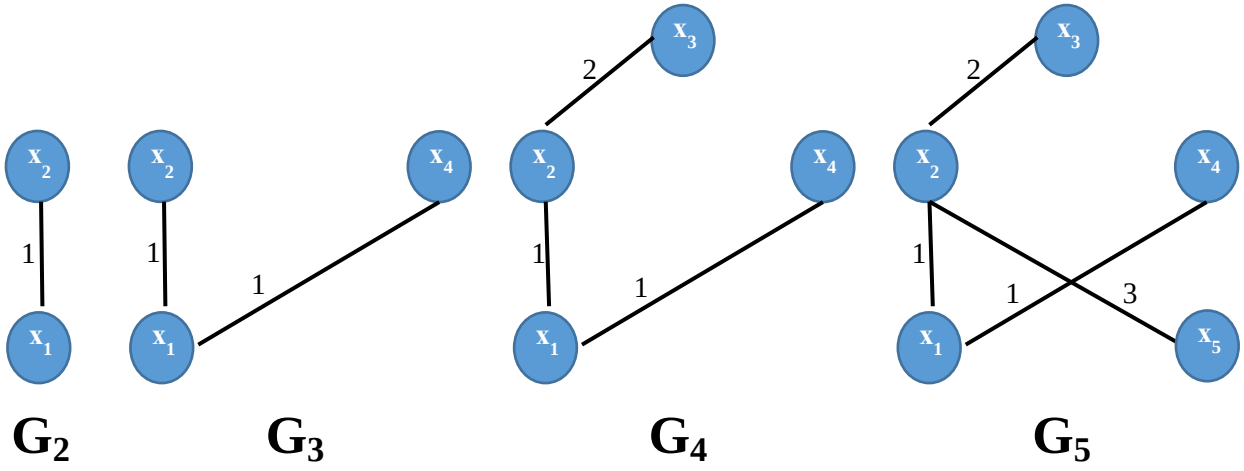
Вместе с этим ребром включаем в $G4$ вершину $x3$, не содержащуюся в $G3$. Полагаем $i = 4$ и переходим к шагу 3.

Шаг 3. Так как $i \neq n$, поэтому переходим к шагу 4.

Шаг 4. Строим граф $G5$, добавляя к графу $G3$ новое ребро минимальной длины из ребер $(x1, x5)$, $(x2, x5)$, $(x4, x5)$. Таких ребер длины три два: $(x2, x5)$ и $(x4, x5)$. Возьмем $(x2, x5)$. Вместе с этим ребром включаем в $G5$ вершину $x5$, не содержащуюся в $G4$. Полагаем $i = 5$ и переходим к шагу 3.

Шаг 3. Так как $i = n$, то граф $G5$ – искомое минимальное остовное дерево. Суммарная длина ребер равна 1 + 1 + 2 + 3 = 7.

Процесс построения минимального остовного дерева изображен на рисунке:

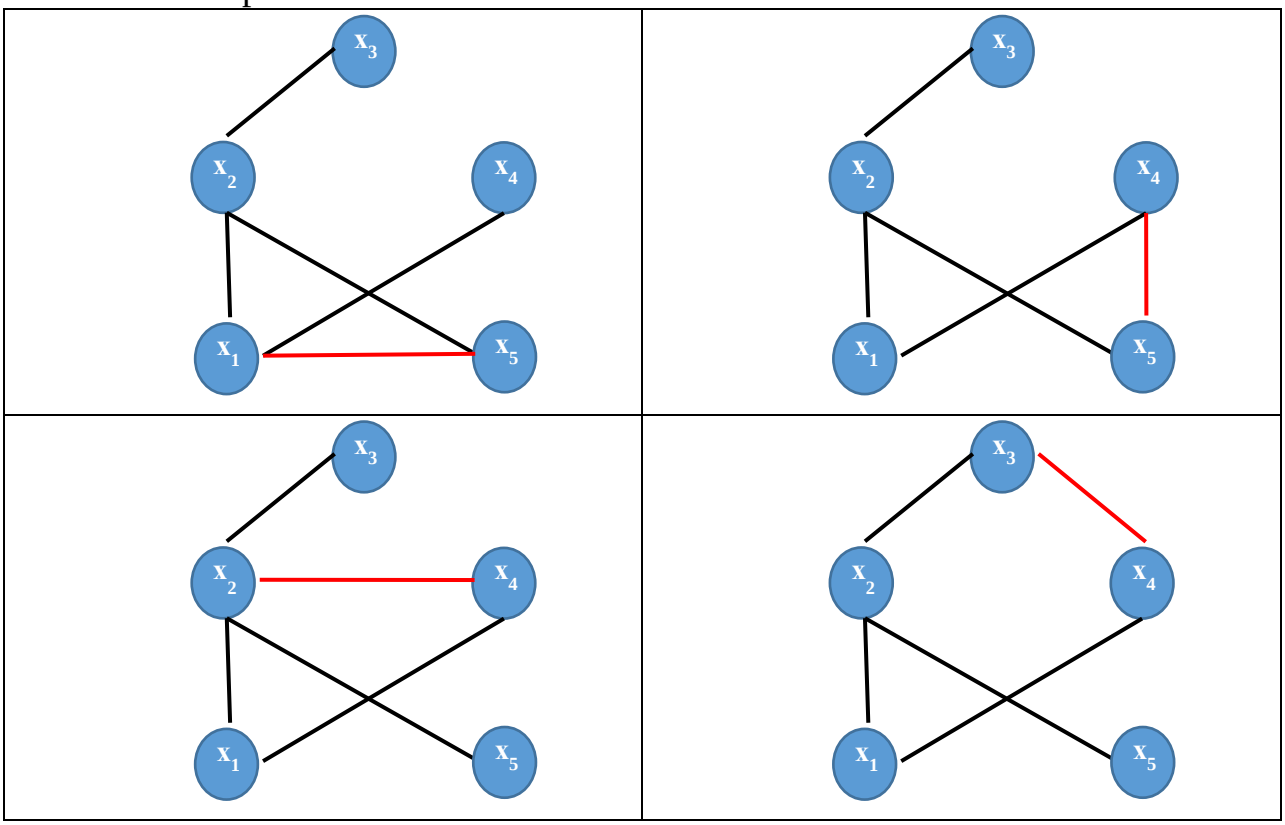


3.4.4.3 Алгоритм построения системы независимых циклов графа

1. Строится произвольный остов графа. В исходном графе отмечаются ребра, включенные в остов.
2. Выбирается очередное неотмеченное ребро и стоится цикл, содержащий это ребро и ребра остова графа. Такой цикл обязательно существует, т.к. вершины остова связаны. Рассмотренное ребро отмечается и если есть еще неотмеченные ребра, то выполняется п.2, иначе – п.3.
3. По формуле Эйлера Υ=m-n+p производится проверка числа построенных циклов (для контроля).

Пример:

Υ=m-n+p=8-5+1=4



3.4.5 Кратчайшая раскраска графа

3.4.5.1 Основные определения

Пусть G = (M, R) – неорграф без петель.

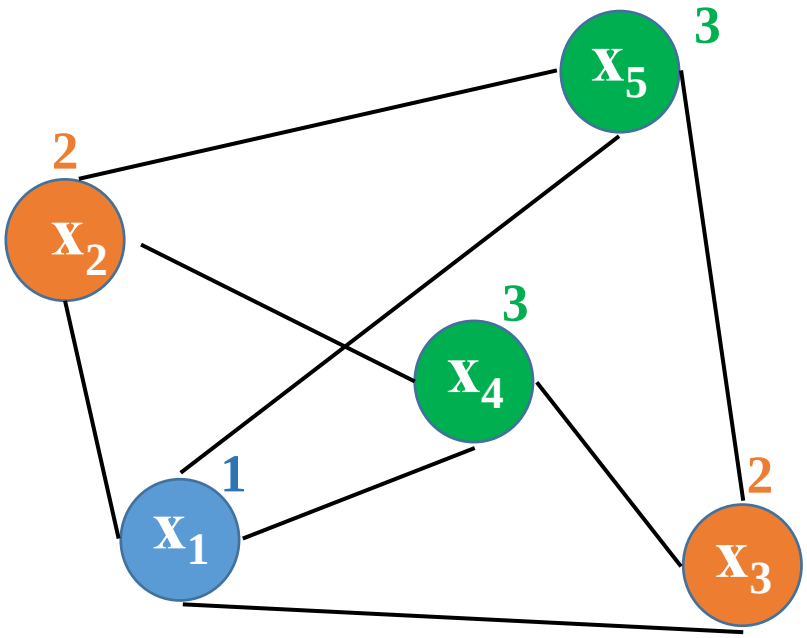
Раскраской (вершин) графа G называется такое задание цветов вершинам G, что если [a, b] – ребро, то вершины a и b имеют различные цвета.

Вместо красок можно использовать числа 0, 1, 2,…

Раскраска называется правильной, если образы любых двух смежных вершин различны.

Хроматическим числом χ (G) графа G называется минимальное число цветов, требующееся для раскраски G.

Пример:



Теорема. Для плоских (планарных) графов хроматическое число $\chi \leq 4$.

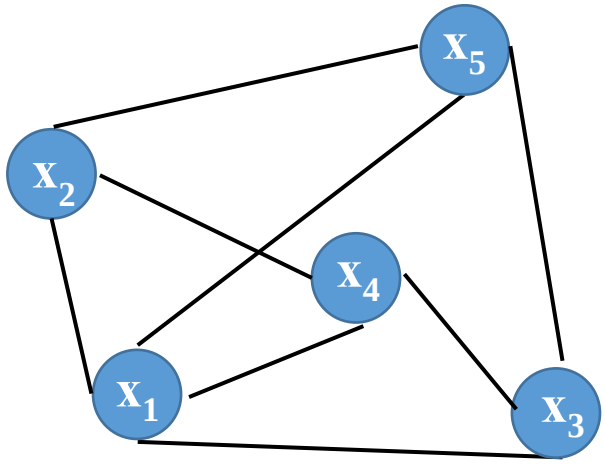
3.4.5.2 Внутренне и внешне устойчивые множества вершин графа

Внутренне устойчивое множество. Подмножество $X_v \subseteq X$ вершин графа $G(X)$ называется внутренне устойчивым, если вершины этого подмножества не смежные, т.е.

$$X_{vt} \cap \left(\bigcup_{x_i \in X_{vt}} G(x_i) \right) = \emptyset$$

Очевидно следует искать максимальные внутренне устойчивые подмножества вершин графа. Эта задача решается посредством применения алгоритма граничного перебора с поиском максимальных совместимых подмножеств по матрице отношений графа или непосредственно по самому графу. Отношение G графа считается отношением противоречия.

Пример:



x_1	x_2	x_3	x_4	x_5	
1↘					$y_1=\{x_1\}$
	1	1↘			$y_2=\{x_2,x_3\}$
	1↘				-
		1↘			-
			1	1↘	$y_3=\{x_4,x_5\}$
				1↘	-

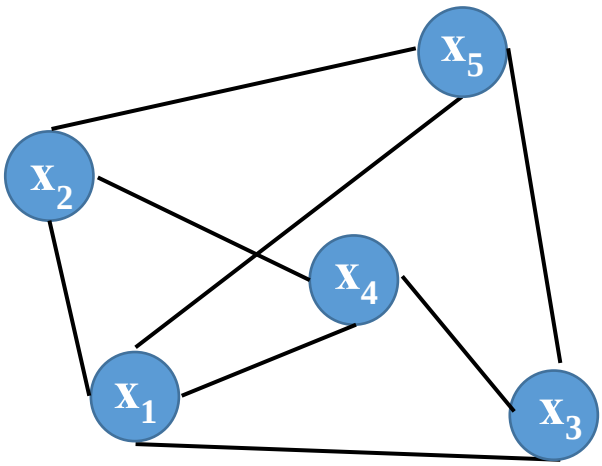
Внешне устойчивое множество. Подмножество $X_{vch} \subseteq X$ вершин графа $G(X)$ называется внешне устойчивым, если для каждой вершины x_i графа удовлетворяется хотя бы одно из условий:

I: $x_i \subseteq X_{vch}$;

II: $x_i \subseteq G^{-1}(x_j)(x_j \in X_{vch})$, *т.е.* $X_{vch} \cup \left(\bigcup_{x_i \in X_{vch}} G^{-1}(x_i)\right) = X$

Эта задача имеет интерпретацию расстановки часовых на вершинах графа с тем, чтобы защитить все вершины графа: вершина x_i защищена в том случае, если на ней стоит часовой $x_i \subseteq X_{vch}$ или если она видна из вершины $x_j \subseteq X_{vch}(x_j \in G^{-1}(x_j))$. Очевидно, необходимо строить минимальные внешние устойчивые множества.

Пример:



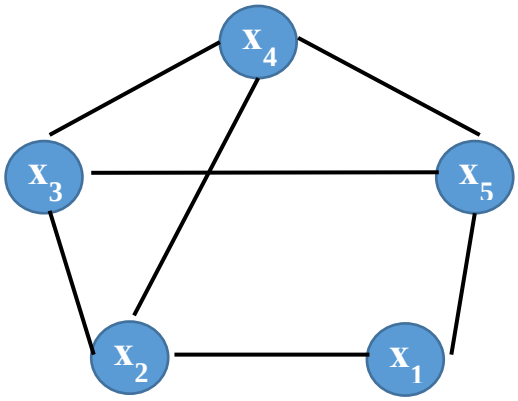
Возможные варианты:
$\{x_1\}$,
$\{x_2,x_3\}$,
$\{x_4,x_5\}$

Алгоритм построения минимальных внешне устойчивых множеств вершин графа:

1. Матрица отношения G графа транспонируется (получается отношение G^{-1}). Диагональные элементы приравниваются к единице (сама вершина себя защищает).
2. Решается задача построения покрытий строками матрицы преобразованного отношения всех ее столбцов.

Пример:

Матрица G:



	x_1	x_2	x_3	x_4	x_5
x_1	0	1	0	0	1
x_2	1	0	1	1	0
x_3	0	1	0	1	1
x_4	0	1	1	0	1
x_5	1	0	1	1	0

Матрица G^{-1}с единицами в диагональных элементах:

	x_1	x_2	x_3	x_4	x_5
x_1	1	1			1
x_2	1	1	1	1	
x_3		1	1	1	1
x_4		1	1	1	1
x_5	1		1	1	1

Решаем задачу нахождения хотя бы одного кратчайшего покрытия:

	x_1	x_2	x_3	x_4	x_5
x_1	1	1			1
x_2	1	1	1	1	
x_3		1	1	1	1
x_4		1	1	1	1
x_5	1		1	1	1

Разложение по столбцу x_1:

$\{x_1,x_2\}$

	x_3
x_2	1
x_3	1
x_5	1

$\{x_2,x_3\}$

	x_5
x_3	1
x_5	1

$\{x_5,x_3\}$

	x_2
x_3	1

Таким образом построены три кратчайших покрытия, которые служат внешне устойчивыми множествами вершин заданного графа. Для поиска всех минимальных внешне устойчивых множеств необходимо решать задачу построения всех безызбыточных покрытий.

3.4.5.3 Алгоритм кратчайшей раскраски графа

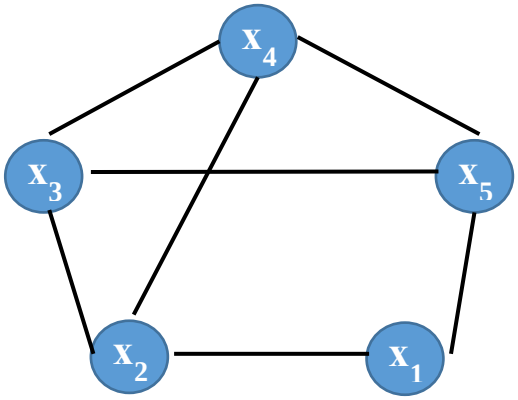
Очевидно, то внутренне устойчивое множество является максимальным (не обязательно наибольшим) тогда и только тогда, когда оно внешне устойчиво. С другой стороны, внешне устойчивое множество не обязательно внутренне устойчиво.

Каждое *максимальное внутренне устойчивое множество* вершин графа может быть *покрашено одной краской*. Для решения задачи необходимо выбрать минимальное число максимальных внутренне устойчивых множеств, содержащих в совокупности все вершины графа, т.е. решить задачу кратчайшего покрытия максимальными внутренне устойчивыми множествами всех вершин графа.

Алгоритм:

1. Методом граничного перебора строятся все максимальные внутренне устойчивые множества вершин графа.
2. Строится таблица покрытий, строками которой являются найденные внутренне устойчивые подмножества, столбцами – вершины графа. Таблица отражает отношение принадлежности вершин максимальных внутренне устойчивых множеств множеству вершин графа.
3. Решается задача построения одного кратчайшего покрытия.
4. Вершины, принадлежащие каждому подмножеству, вошедшему в найденное кратчайшее покрытие, окрашиваются одной краской. Если некоторая вершина принадлежит нескольким вошедшим в покрытие подмножествам, то она в одном остается, а из остальных исключается.

Пример:



x_1	x_2	x_3	x_4	x_5	
1		1↘			$\{x_1,x_3\}$
1			1↘		$\{x_1,x_4\}$
1↘					-
	1			1↘	$\{x_2,x_5\}$
	1↘				
		1↘			-
			1↘		-
				1↘	-

Таблица покрытий: покрытие A, B, C.

	x_1	x_2	x_3	x_4	x_5
A	1		1		
B	1			1	
C		1			1

Раскраска:

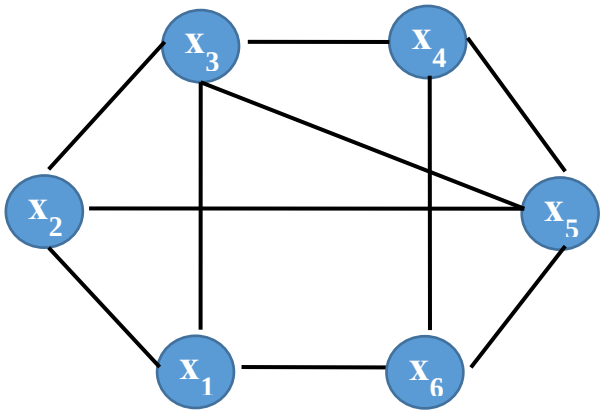
A = $\{x_1,x_3\}$ – краска 0

B = $\{x_4\}$ – краска 1

C = $\{x_2,x_5\}$ – краска 2

Пример:

Методом граничного перебора строятся все максимальные внутренне устойчивые множества вершин графа

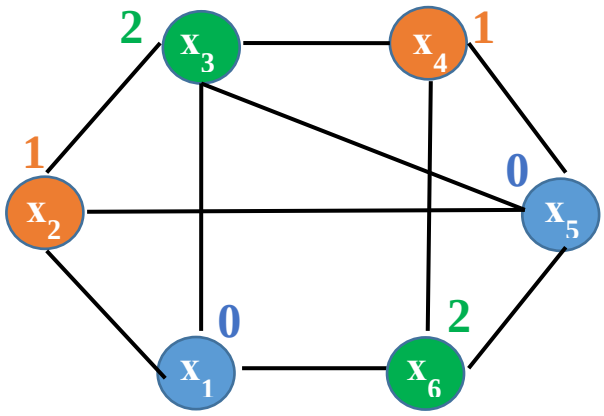


x_1	x_2	x_3	x_4	x_5	x_6	
1			1↘			$\{x_1,x_4\}$
1				1↘		$\{x_1,x_5\}$
1↘						-
	1		1↘			$\{x_2,x_4\}$
	1				1↘	$\{x_2,x_6\}$
	1↘					-
		1			1↘	$\{x_3,x_6\}$
		1↘				-
			1↘			-
				1↘		-
					1↘	-

Таблица покрытий: B, C, E.

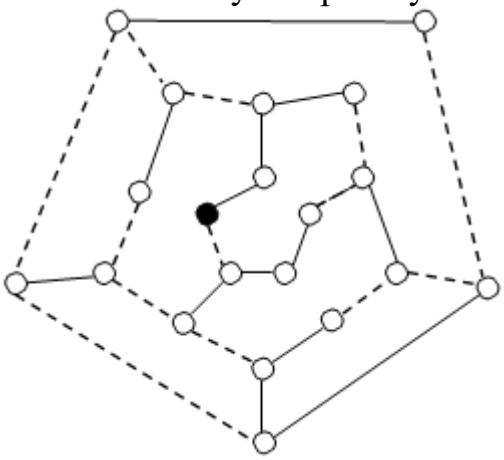
	x_1	x_2	x_3	x_4	x_5	x_6
A	1			1		
B	1				1	
C		1		1		
D		1				1
E			1			1

Покрытие - B, C, E.
B - $\{x_1,x_5\}$ – краска 0
C - $\{x_2,x_4\}$ – краска 1
E - $\{x_3,x_6\}$ – краска 2



3.4.6 Гамильтонов контур

Гамильтоновый цикл и контур связаны с именем Уильяма Гамильтона (Hamilton W.), который в 1859 году предложил следующую игру-головоломку: требуется, переходя по очереди от одной вершины додекаэдра к другой вершине по его ребру, обойти все 20 вершин по одному разу и вернуться в начальную вершину.



3.4.6.1 Постановка задачи о гамильтоновом контуре минимальной стоимости

Гамильтонов контур (ГК) – замкнутый путь по вершинам графа.

Пусть задан орграф (G,X), дуги которого взвешены функцией $l(x_i, x_j)$.

Необходимо составить **(ГК)**, который удовлетворяет следующим условиям:

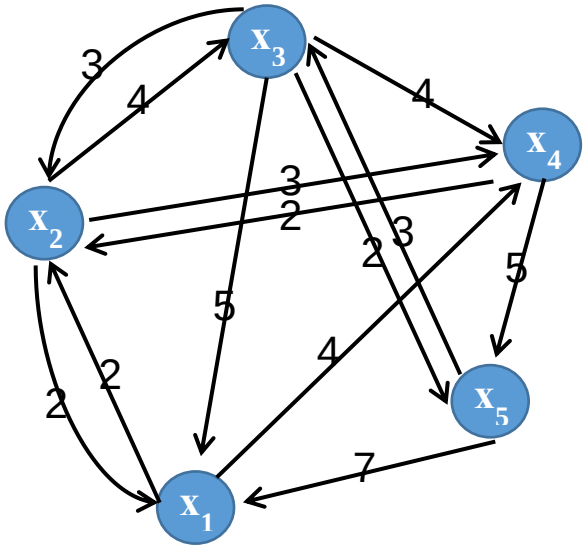
1. контур, если такой существует, включает в себя все вершины графа;
2. для каждой вершины только одна дуга входа этой вершины принадлежит контуру и только одна дуга исхода вершины принадлежит контуру (каждая вершина включена в контур только один раз);

3. сумма f весов дуг построенного контура должна быть минимальной (т.к. строится ГК минимальной стоимости):

$$f = \sum_{(x_i,x_j)\in ГК} (x_i, x_j) \to min$$

Такой ГК называется **минимальным**.

Пример:



Возможные гамильтоновы контуры:

I: $(x_1, x_2)(x_2, x_3)(x_3, x_4)(x_4, x_5)(x_5, x_1)$

$$f = 2 + 4 + 4 + 5 + 7 = 22$$

II: $(x_1, x_2)(x_2, x_4)(x_4, x_5)(x_5, x_3)(x_3, x_1)$

$$f = 2 + 3 + 5 + 3 + 5 = 18$$

III: $(x_1, x_4)(x_4, x_2)(x_2, x_3)(x_3, x_5)(x_5, x_1)$

$$f = 4 + 2 + 4 + 2 + 7 = 19$$

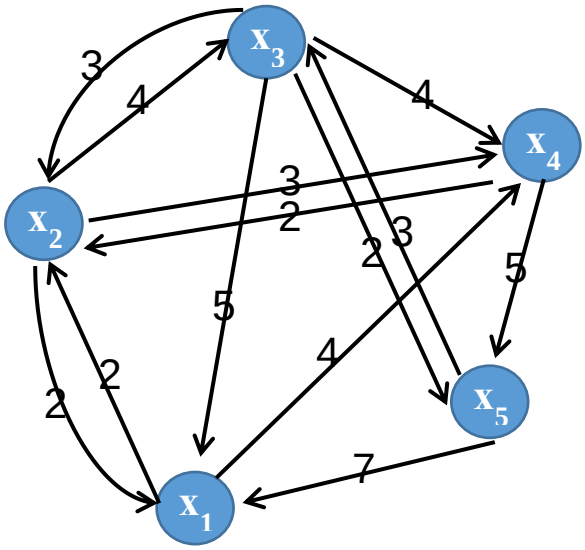
Возможные интерпретации задачи о ГК:

– выбор расписания проведения работ;
– быстрейший путь сбора подписей;
– задача коммивояжера

3.4.6.2 Метод ветвей и границ для построения гамильтонова контура минимальной стоимости

Исходным служит представление графа в виде матрицы весов (отношение $l(x_i, x_j)$). Если дуги (x_i, x_j) в графе нет, то $l(x_i, x_j) = \infty$.

Пример:



Для заданного графа матрица весов $l(x_i, x_j)$:

	x_1	x_2	x_3	x_4	x_5
x_1	∞	2	∞	4	∞
x_2	2	∞	4	3	∞
x_3	5	3	∞	4	2
x_4	∞	2	∞	∞	5
x_5	7	∞	3	∞	∞

Вычисление оценки.

ГК содержит только по одной дуге, принадлежащей $G^{+1}x_i$ и $G^{-1}x_i$ – входу и исходу этой вершины. При вычислении функции качества ГК только одно число из каждой строки матрицы $l(x_i, x_j)$ будет входить в сумму весов ребер ГК.

Пусть $l(x_i) = \min_{x_j} l(x_i, x_j)$ – минимальное число в строке. Преобразуем матрицу $|l(x_i, x_j)|$ в $|l'(x_i, x_j)| = |l(x_i, x_j) - l(x_i)|$.

Хотя в каждой строке из всех ее элементов вычислена величина $l(x_i)$, но поскольку в функцию качества ГК входит только одно из них, то справедливо утверждение, что функция качества ГК, полученного по исходной матрице

$$f = f' + \sum_{i=1}^{n} l(x_i) = f' + \xi_{\text{вых}}$$

Аналогично проводится вычисление для столбцов:

$l'(x_j) = \min_{x_i} l'(x_i, x_j)$;

$|l''(x_i, x_j)| = |l'(x_i, x_j) - l'(x_j)|$;

$$\xi_{\text{вх}} = \sum_{j=1}^{n} l'^{(j)}$$

$\xi = \xi_{\text{вых}} + \xi_{\text{вх}}; \quad f = f'' + \xi.$

Пример:

	x_1	x_2	x_3	x_4	x_5	$l(x_i)$
x_1	∞	2	∞	4	∞	2
x_2	2	∞	4	3	∞	2
x_3	5	3	∞	4	2	2
x_4	∞	2	∞	∞	5	2
x_5	7	∞	3	∞	∞	3
						/11

	x_1	x_2	x_3	x_4	x_5	
x_1	∞	0	∞	2	∞	
x_2	0	∞	2	1	∞	
x_3	3	1	∞	2	0	
x_4	∞	0	∞	∞	3	
x_5	4	∞	0	∞	∞	
$l(x_j)$	0	0	0	1	0	**/1**

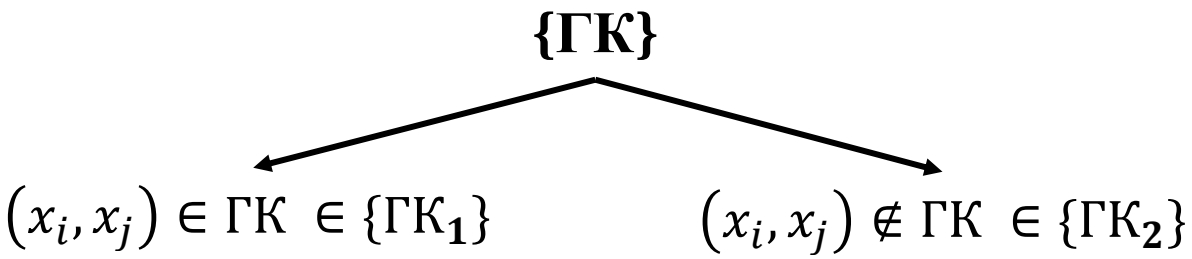
	x_1	x_2	x_3	x_4	x_5
x_1	∞	0	∞	1	∞
x_2	0	∞	2	0	∞
x_3	3	1	∞	1	0
x_4	∞	0	∞	∞	3
x_5	4	∞	0	∞	∞

$|l(x_i, x_j)|$, $\xi_{\text{вых}} = 11$, $|l'(x_i, x_j)|$, $\xi_{\text{вх}} = 1$, $|l''(x_i, x_j)|$,
$\xi = \xi_{\text{вых}} + \xi_{\text{вх}}$=11+1=12,
$f = f'' + 12$.

По крайней мере, функция f качества любого ГК матрицы $|l(x_i, x_j)|$ больше или равна ξ, таким образом, величина ξ может служить оценкой качества возможных ГК по матрице $|l(x_i, x_j)|$.

Идея ветвления.

По любой дуге можно разложить множество {ГК} на два непересекающихся подмножества: {ГК$_1$}, содержащие данную дугу, и {ГК$_2$}, не содержащие ее.



Одно или оба подмножества могут быть пустыми.

Используется усиление алгоритма, приводящее к сокращению процесса решения: нужно стараться выбрать дугу так, чтобы во множестве {ГК$_1$} были ГК меньшей длины, а во множестве {ГК$_2$} – большей длины, тогда множество {ГК$_2$} возможно будет не рассматривать.

Для матрицы $|l''(x_i, x_j)|$ выбор дуги (x_i, x_j), для которой $l(x_i, x_j) = 0$ не приводит непосредственно к увеличению длины ГК. Значит, можно рассматривать такие дуги. Их в матрице $|l''(x_i, x_j)|$ много, по крайней мере по одной в строке и столбце.

Запрещение дуги должно приводить к увеличению оценки длины ГК на возможно большую величину $\theta(x_i, x_j)$. Запрещение дуги (x_i, x_j) равносильно тому, что $l(x_i, x_j) = \infty$, но тогда в строке x_i и столбце x_j один из нулей исчезает, значит можно пересмотреть оценку

$$\theta(x_i, x_j) = \min_{x_k \neq x_j} l(x_i, x_k) + \min_{x_k \neq x_i} l(x_k, x_j)$$

Выбирается та дуга (x_k, x_l), для которой $\theta(x_i, x_j)$ →max.

Пример:

	x_1	x_2	x_3	x_4	x_5
x_1	∞	0	∞	1	∞
x_2	0	∞	2	0	∞
x_3	3	1	∞	1	0
x_4	∞	0	∞	∞	3
x_5	4	∞	0	∞	∞

$$\theta(x_1, x_2) = l(x_1, x_4) + l(x_4, x_2) = 1 + 0 = 1,$$
$$\theta(x_2, x_1) = l(x_2, x_4) + l(x_3, x_1) = 0 + 3 = 3,$$
$$\theta(x_2, x_4) = l(x_2, x_1) + l(x_1, x_4) = 0 + 1 = 1,$$
$$\theta(x_3, x_5) = l(x_3, x_2) + l(x_4, x_5) = 1 + 3 = 4,$$
$$\theta(x_4, x_2) = l(x_4, x_5) + l(x_1, x_2) = 3 + 0 = 3,$$
$$\theta(x_5, x_3) = l(x_5, x_1) + l(x_2, x_3) = 4 + 2 = 6,$$

Вычислим $\theta(x_k, x_l) = \max(\theta(x_i, x_j)) = \theta(x_5, x_3) = 6$. Оценки $\theta(x_i, x_j)$ можно непосредственно изображать на матрице $l''(x_i, x_j)$:

	x_1	x_2	x_3	x_4	x_5
x_1	∞	0_1	∞	1	∞
x_2	0_3	∞	2	0	∞
x_3	3	1	∞	1	0_4
x_4	∞	0_3	∞	∞	3
x_5	4	∞	0_6	∞	∞

Ветвление можно вести по дуге (x_5, x_3):

ГК=Ø, $|l''(x_i, x_j)|$**;** ξ **=12**

$(x_5, x_3) \in$ ГК	ГК $= \emptyset$
$\|l^*(x_i, x_j)\|$	$\|l^{**}(x_i, x_j)\|$
$\xi^* = \xi + l(x_5, x_3) = 12+0=12$	$\xi^{**} = \xi + \theta(x_i, x_j) = 12+6=18$

Проведение ветвления по дуге

По любой дуге (x_k, x_l). Матрица $|l^*(x_i, x_j)|$ строится по матрице $l''(x_i, x_j)$:

a) удаляются строка x_k и столбец x_l;

b) чтобы не допускать замыкания раньше, чем будут пройдены все вершины, запрещается замыкание всех путей, вошедших в выбранную часть ГК (удаляется дуга, соединяющая конечную вершину пути с начальной);

c) в полученной матрице $|l^*(x_i, x_j)|$ строится оценка $\xi^*_{\text{доп}} = \xi^*_{\text{вых}} + \xi^*_{\text{вх}}$ и матрица преобразуется до $|l^{*\prime\prime}(x_i, x_j)|$ точно так же, как преобразовывалась матрица $|l(x_i, x_j)|$ до $|l''(x_i, x_j)|$

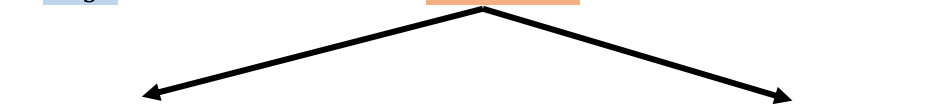
Матрица $|l^{**}(x_i, x_j)|$ получается заменой в матрице $|l''(x_i, x_j)|$ величины $l''(x_k, x_l) = 0$ на $l^{**}(x_k, x_l) = \infty$.

Для обеих матриц $|l^*(x_i, x_j)|$ и $|l^{**}(x_i, x_j)|$ проводится оценка и дополнительные значения $\xi^*_{\text{доп}}$ и $\theta(x_k, x_l)$ прибавляются к ξ, матрицы $|l^*|$ и $|l^{**}|$ преобразуются к виду $|l^{*\prime\prime}|$ и $|l^{**\prime\prime}|$.

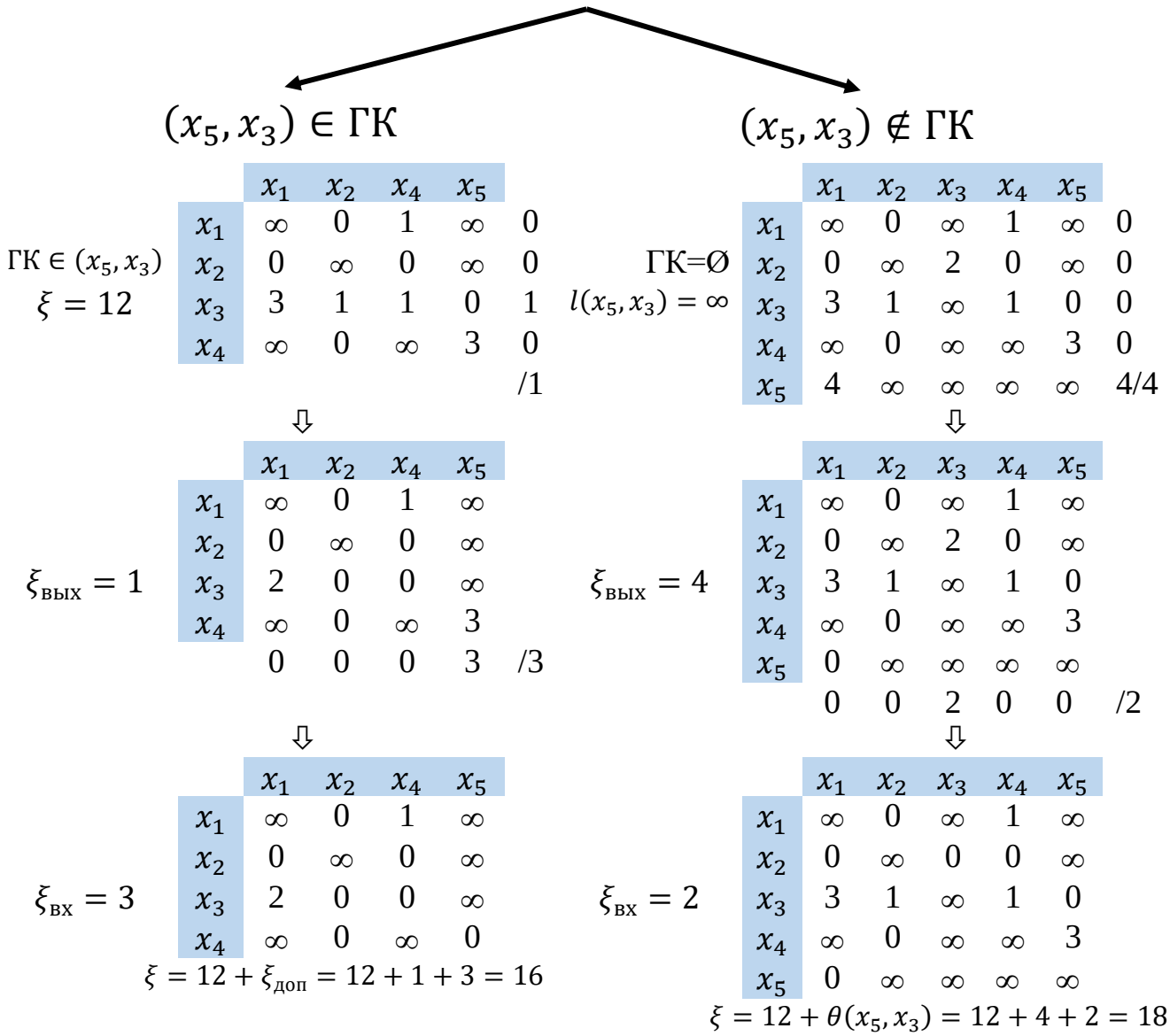
Продолжим разложение для каждого примера. Величины $\theta(x_i, x_j)$ для дуг с $l(x_i, x_j) = 0$ записываем в соответствующие элементы в качестве нижнего индекса:

ГК=Ø
$\xi = 12$

	x_1	x_2	x_3	x_4	x_5
x_1	∞	0_1	∞	1	∞
x_2	0_3	∞	2	0	∞
x_3	3	1	∞	1	0_4
x_4	∞	0_3	∞	∞	3
x_5	4	∞	0_6	∞	∞



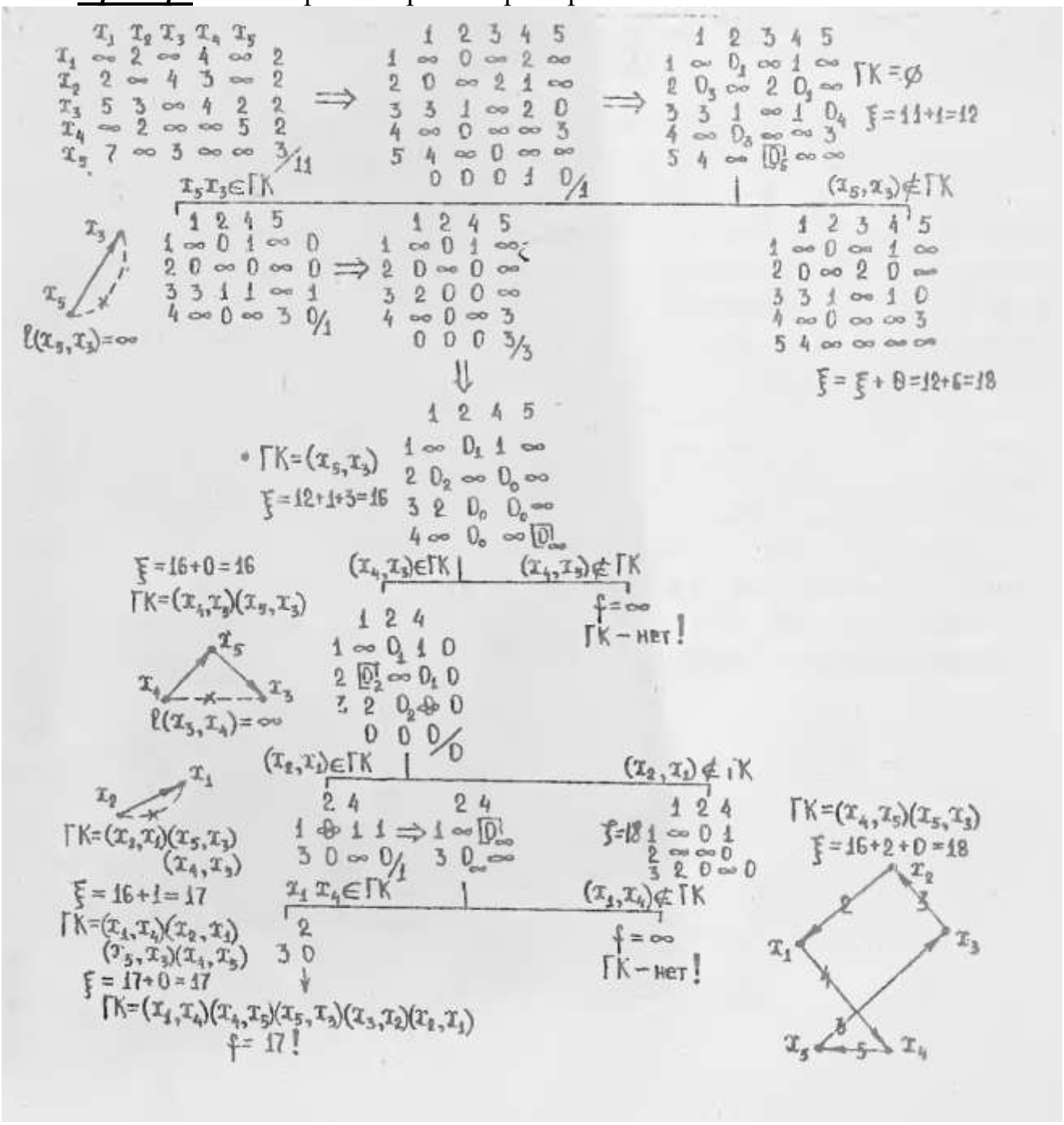
$(x_5, x_3) \in$ ГК $\qquad$ $(x_5, x_3) \notin$ ГК



Получение решения.

Т.к. каждой вершине дерева приписывается множество дуг, включенных в ГК, то на висячей вершине дерева решения, для которой матрица $|l^*(x_i, x_5)|$ оказывется пустой, множество ГК с точностью до упорядочения дуг составит ГК и оценка ξ будет уже функцией f – длиной ГК.

Пример: Полное рассмотрение примера….



3.4.6.3 Краткая формулировка алгоритма

I. От исходной матрицы $|l|$ переходим к матрице $|l''|$ и оценке ξ, ГК=Ø. Тройку $|l''|$, ξ и ГК помещаем в корень дерева. Рассматриваем вершину – корень дерева.

II. В рассматриваемой вершине дерева строим оценки $\theta(x_i, x_j)$ для дуг, для которых $l''(x_i, x_j) = 0$ и выбираем дугу (x_k, x_l) с наибольшей оценкой $\theta(x_k, x_l)$.

III. Проводим разложение матрицы $|l''|$ на дуге (x_k, x_l) на $|l^*|$ и $|l^{**}|$, уточняем оценки ξ^* и ξ^{**} и получаем матрицы $|l^{*\prime\prime}|$ и $|l^{**\prime\prime}|$.

Если ξ^* или ξ^{**} равны ∞, то обрываем соответствующую ветвь.

Если ξ^* или ξ^{**} больше стоимости $f_л$ лучшего из найденных ГК, то также обрываем соответствующую ветвь.

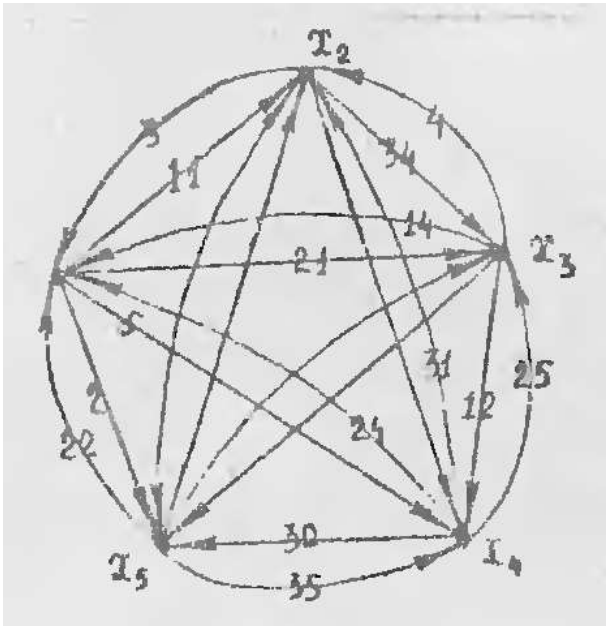
Если $|l^*| = \emptyset$, то на этой вершине получим ГК и f.

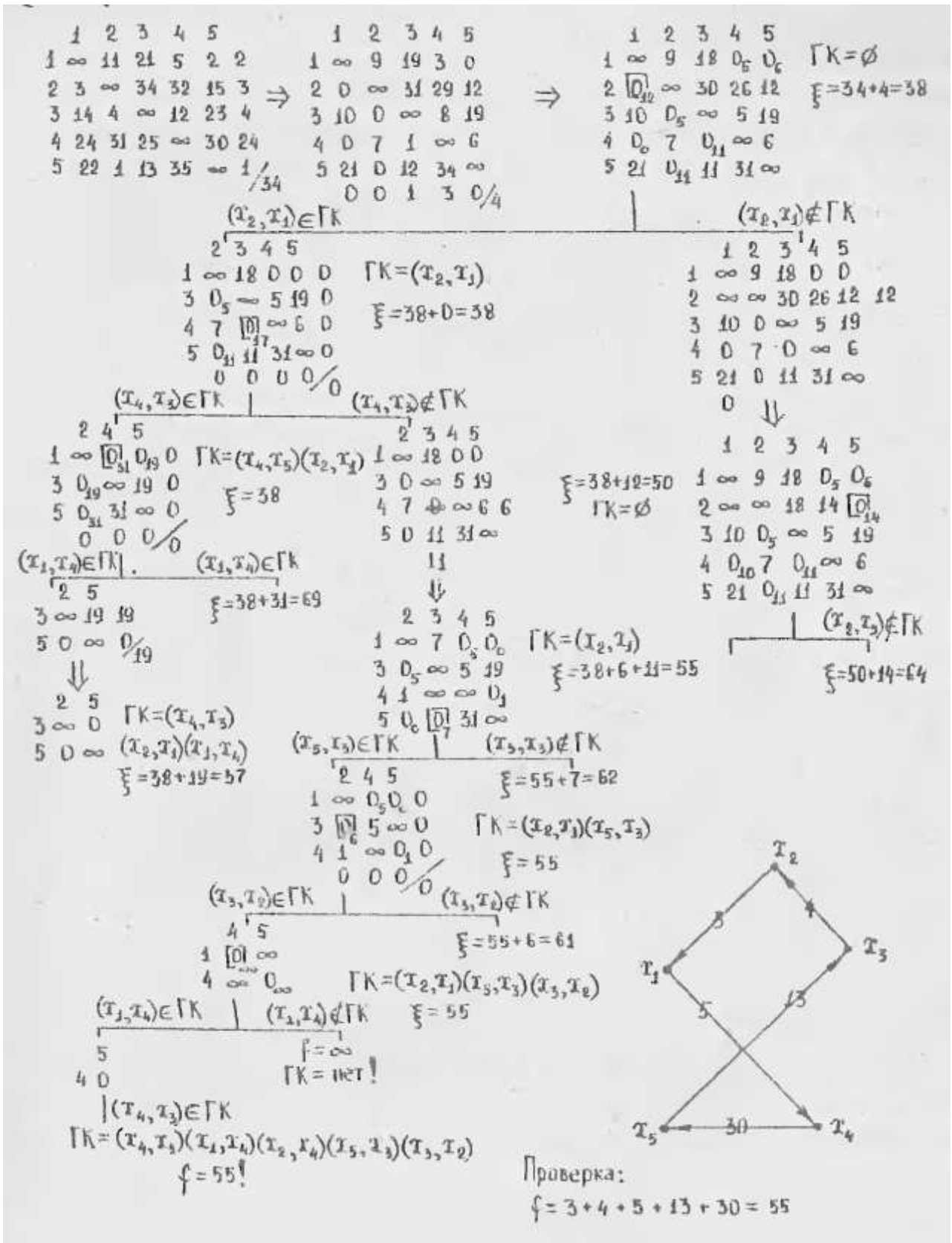
Сравним f и $f_л$. Если $f < f_л$, то запоминаем f как стоимость $f_л$ лучшего результата, иначе обрываем соответствующую ветвь. Обрываем все концевые вершины, где $\xi \geq f_л$.

IV. Если есть висячие вершины с непустой матрицей $|l''|$, то выбираем из них одну с меньшим значением ξ и выполняем п.2, иначе процесс заканчивается; решение, соответствующее значению $f_л$, является гамильтоновым контуром минимальной длины.

Замечание: Весь процесс изображается в виде дерева во всех вариантах звездочки *, ** и штрихи $',''$ не ставятся.

Пример:




ГК=∅
ξ=34+4=38
ГК=(T2,T3)
ξ=38+0=38
ГК=(T4,T5)(T2,T3)
ξ=38
ξ=38+31=69
ГК=(T4,T3)
(T2,T1)(T1,T4)
ξ=38+19=57
ξ=38+12=50
ГК=∅
ξ=50+14=64
ГК=(T2,T3)
ξ=38+6+11=55
ξ=55+7=62
ГК=(T2,T3)(T5,T3)
ξ=55
ξ=55+6=61
ГК=(T2,T3)(T5,T3)(T3,T2)
ξ=55
ГК=(T4,T3)(T1,T4)(T2,T4)(T5,T3)(T3,T2)
ξ=55!
T2
T3
T1
T5
T4
30
Проверка:
ξ=3+4+5+13+30=55

3.4.7 Потоки в сети

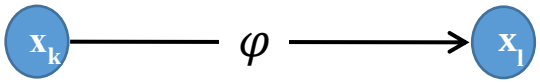
3.4.7.1 Определение потока в сети

Пусть (G,X) – ориентированный граф, в котором определены две вершины: **вершина истока** $x_{is} \in X$ и **вершина стока** $x_{st} \in X$. **Поток** вытекает из вершины x_{is}, проходит по вершинам сети и собирается в вершине x_{st}.

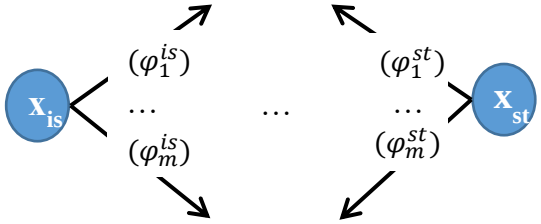
Поток в сети формируется из потоков по каждой дуге сети. Допускаются только положительные значения потока $\varphi(g)$ по каждой дуге:

$$g \in G\ (\varphi(g) \geq 0) \qquad (5.1)$$

Поток по дуге обычно обозначается числом в скобках.



Сумма потоков, вытекающих из вершины x_{is}, равна сумме потоков, приходящих к вершине x_{st}.



$$V = \sum_{i=1}^{m} \varphi_i^{is} = \sum_{i=1}^{s} \varphi_i^{st} \qquad (5.2)$$

Величина V – **оценка потока в сети**. Условие (5.2) должно соблюдаться для всех внутренних вершин сети: сумма потоков, приходящих к вершине $x_l (l \neq is, st)$, равна сумме потоков, вытекающих из вершины x_l:

$$\sum \varphi(x_i(\forall x_i \in G^{-1}(x_l)), x_l) = \sum \varphi(x_l(\forall x_i \in G^{+1}(x_l)), x_i) \qquad (5.3)$$

Абстрагируясь от интерпретаций, **поток в сети** – это функция $\varphi(g)$, определенная на дугах $g \in G$ сети, которая удовлетворяет условиям (5.1) – (5.3).

В дальнейшем для наглядности будем упорядочивать изображение сети следующим образом: вершину истока помещаем на рисунках слева, а вершину стока – справа, промежуточные вершины – между ними, упорядочивая их по ярусам.

3.4.7.2 Задача о максимальном потоке

Постановка задачи

На дугах $g \in G$ сети определяет функция $C(g) \geq 0$ – пропускная способность дуги g. Ставится задача построения потока в сети, для которого удовлетворяется условие

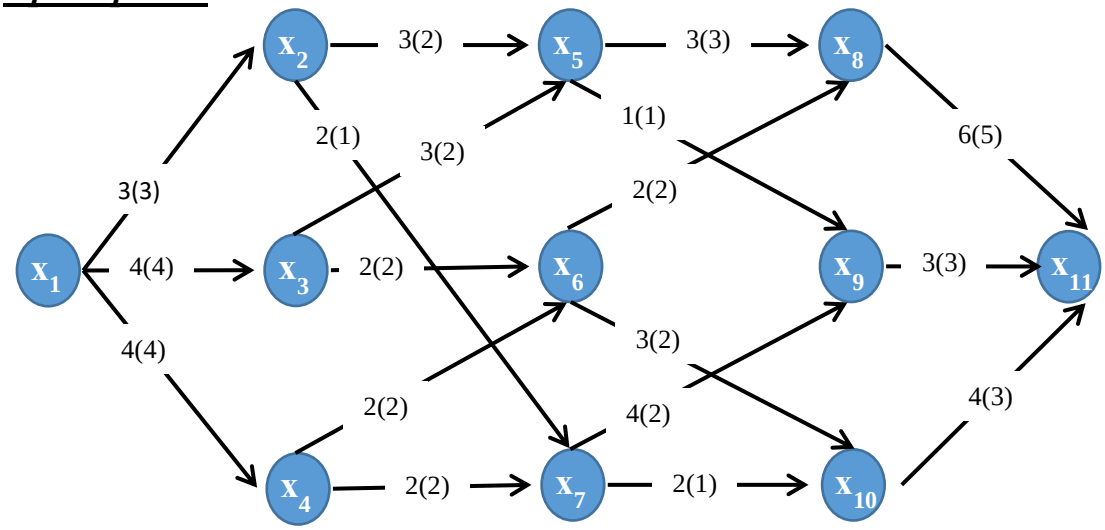
$$\varphi(g) \leq C(g) \tag{5.4}$$

и наибольшее значение принимает оценка V, т.е.

$$V = \sum_{g \in G^{+1}(x_{is})} \varphi(g) \to max \tag{5.5}$$

Пропускная способность $C(g)$ ограничивает допустимый наибольший поток по каждой дуге. Величины $C(g)$ изображаются на дугах сети как числа без скобок. Поскольку поток в сети должен удовлетворять условиям (5.1) – (5.3), то не на всех дугах он одновременно может достигать своего наибольшего значения.

Пример 5.13:



Поток в данной сети очевидно максимален: по каждой из дуг исхода вершины (x_1) поток достигает наибольшего возможного значения, так как равен пропускной способности этих дуг.

В данном примере дуга (x_2, x_5) не достигает своего наибольшего возможного значения: наибольшее суммарное значение потока, приходящего к вершине x_2, $C(x_1, x_2) = 3$, суммарное значение пропускных способностей дуг, исходящих из вершины x_2, равно 5 $(C(x_2, x_5) + C(x_2, x_7) = 3 + 2 = 5)$, но суммарное значение потоков, исходящих из вершины x_2, не может быть больше 3 (условие (5.3)).

Построение полного потока

Процесс построения удобно разбить на два этапа: построение полного потока и коррекция потока до максимального.

Дуга называется **насыщенной**, если $\varphi(g) = C(g)$.

Насыщенный путь - путь от вершины истока до вершины стока который содержит хотя бы одну насыщенную дугу.

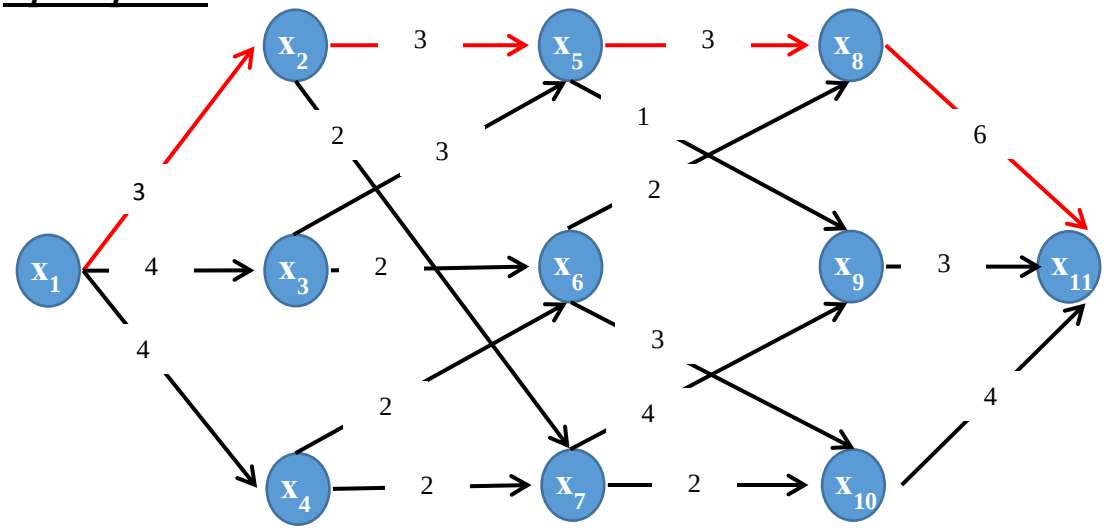
Поток называется **полным**, если все пути в сети от вершины истока до вершины стока насыщены.

При упорядоченном изображении сети процесс нахождения путей от вершины истока до вершины стока достаточно нагляден, поэтому и процесс построения полного потока в сети прост: находится очередной путь $(x_{is} \to x_{st})$, состоящий только из ненасыщенных дуг и по всем дугам g этого пути поток увеличивается на величину $\Delta\varphi$, равную наименьшей разности $C(g) - \varphi(g)$:

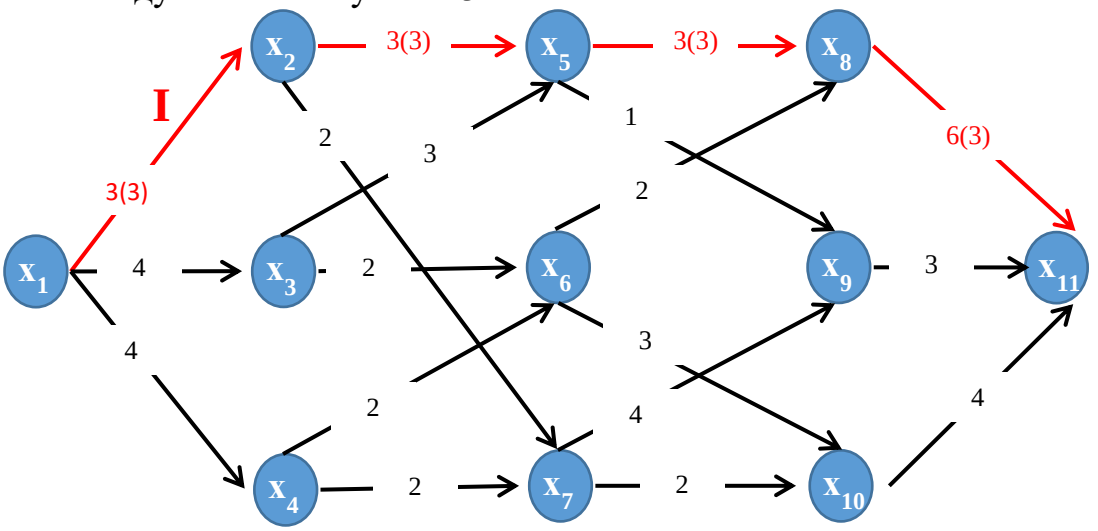
$$\Delta\varphi = \min_{g \in (x_{is} \to x_{st})} C(g) - \varphi(g) \qquad (5.6)$$

Для организации перебора путей $(x_{is} \to x_{st})$ каждый раз выбирается возможное верхнее ребро очередной вершины, к которой подходит путь. Так как необходимо увеличить на всем пройденном пути поток по дугам на одну и ту же величину $\Delta\varphi$, то условия (5.1)-(5.3) в данном алгоритме выполняются автоматически.

Пример 5.14:

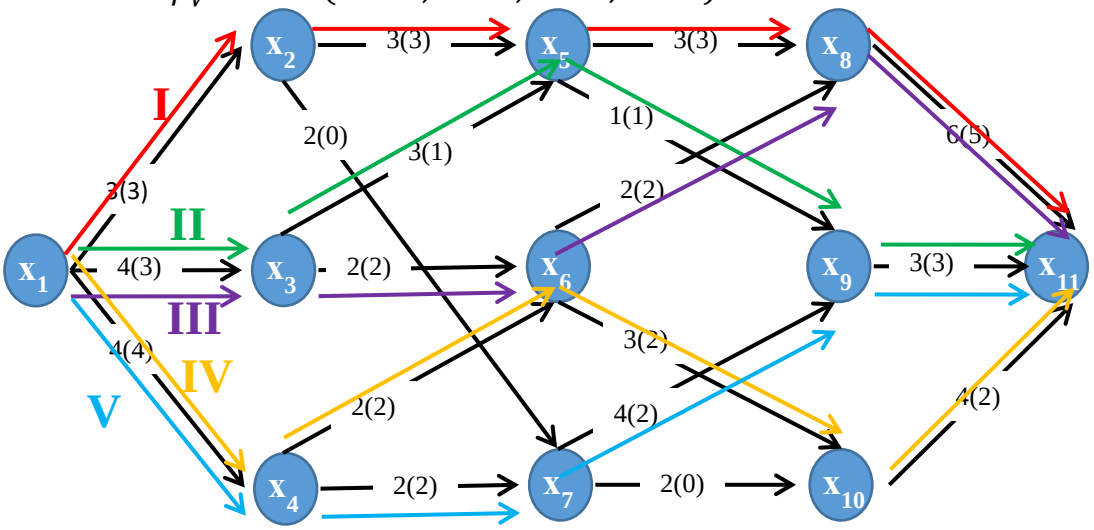


Для выделенного пути $(x_1 \to x_{11}) = (x_1, x_2)(x_2, x_5)(x_5, x_8)(x_8, x_{11})$ величина $\Delta\varphi = \min(3 - 0{,}3 - 0{,}3 - 0{,}6 - 0) = 3$, поэтому увеличиваем поток по всем дугам этого пути на 3.



Очевидно, ненасыщенных путей, содержащих дугу (x_1, x_2), нет, поскольку уже сама эта дуга насыщена, поэтому последовательно строим пути II, III, с дугой (x_1, x_3), IV,V с дугой (x_1, x_4):

$$\Delta\varphi_{II} = \min(4 - 0{,}3 - 0{,}1 - 0{,}3 - 0) = 1;$$
$$\Delta\varphi_{III} = \min(4 - 1{,}2 - 0{,}2 - 0{,}6 - 3) = 2;$$
$$\Delta\varphi_{IV} = \min(4 - 0{,}2 - 0{,}3 - 0{,}4 - 0) = 2;$$
$$\Delta\varphi_{V} = \min(4 - 2{,}2 - 0{,}4 - 0{,}3 - 1) = 2.$$



Поток, построенный в рассмотренном примере, полон. Однако он не максимален, т.к. оценка полученного потока $V = 10$, а оценка возможного потока $V = 11$ (исходный рисунок).

Итак, ***не каждый полный поток максимален***. Дело в том, что мы совершенно произвольно находили ненасыщенные пути и увеличивали поток по этим путям, не заботясь о других возможных вариантах изменения потока, и таким образом «портили» возможное увеличение потока по другим путям. Это обычная ситуация, когда, последовательно решая задачи с оптимизацией решения на отдельном шаге (а не решения в целом), решение может зайти «в тупик» и не получить общего наилучшего решения.

Коррекция произвольного потока до максимального

Алгоритм Форда - Фалкерсона состоит из разметки вершин, выбора цепочки дуг для коррекции потока и самой коррекции потока.

Алгоритм разметки вершин и выбора цепочки дуг для коррекции потока

1. Все вершины сети считаются непомеченными. Вершина истока помечается индексом +0.

2. Выбирается очередная помеченная, но еще не рассмотренная, вершина x_l:

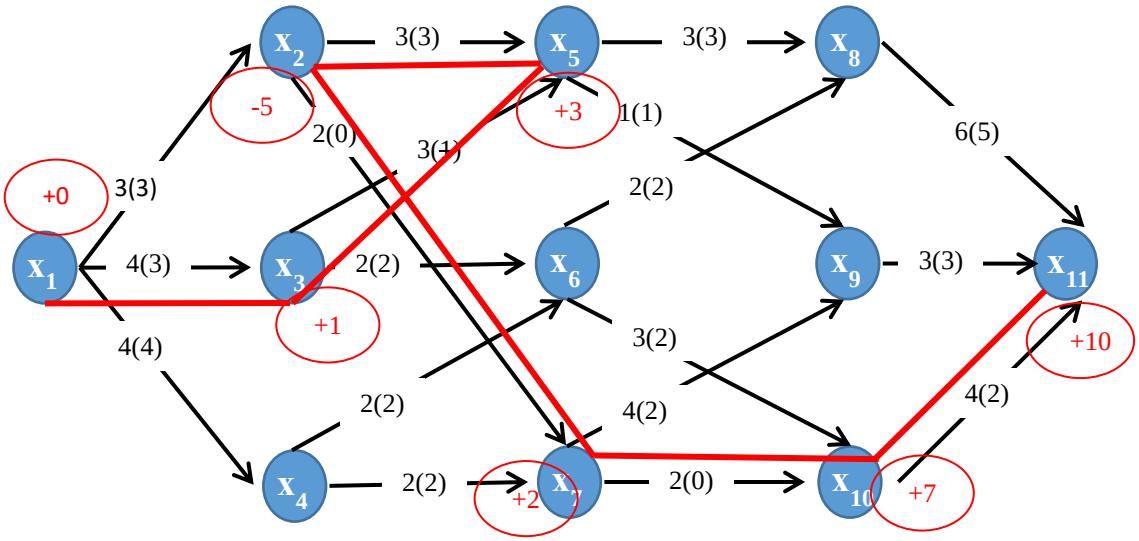
– индексом «+*l*» помечаются непомеченные вершины x_s, достигаемые из вершины x_l по дуге (x_l, x_s), для которой $\varphi(x_l, x_s) \le C(x_l, x_s)$, т.е. дуге, по которой можно увеличить поток из вершины x_l;

– индексом «-*l*» помечаются непомеченные вершины x_k, из которых можно достичь вершину x_l по дуге (x_l, x_k), для которой $\varphi(x_k, x_l) > 0$, т.е. по дуге, по которой можно уменьшить поток в вершину x_l с тем, чтобы направить его по другому пути к вершине стока и таким образом скорректировать поток в сети.

3. Если помечается вершина стока, то процесс расстановки пометок прекращается, иначе, если помеченные, но не рассмотренные вершины, то выполняется п.2; иначе, если все помеченные вершины рассмотрены, но не достигнута вершина стока, делается вывод, что поток максимален.

В формулировке алгоритма изложена идея коррекции: увеличить поток из вершины x_s к вершине x_l, а часть прежнего потока к вершине x_l вернуть и направить по новому пути.

Пример:



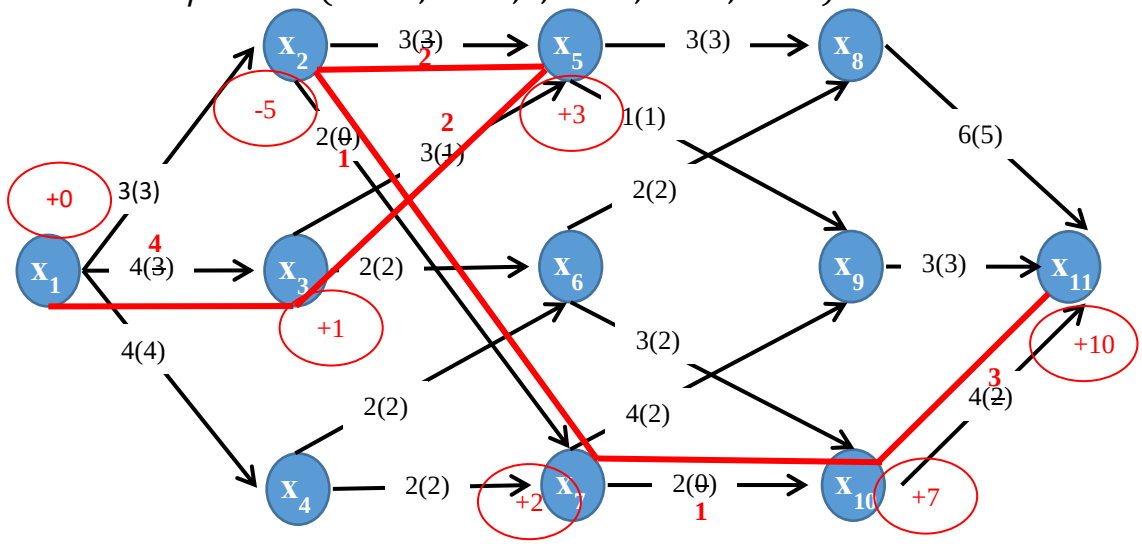
В предложенном примере расставлены индексы «$\pm l$», и помеченной оказалась вершина стока, значит поток в графе не максимален и следует найти цепочку дуг, по которым нужно скорректировать поток. Выделить эту цепочку не сложно, т.к. индексы ($\pm l$) вершин, просматриваемые, начиная с вершины сток, помогают в этом.

Величина $\Delta\varphi$ изменения потока для выделенной цепочки дуг равна:

$$\Delta\varphi = min\begin{cases} C(g) - \varphi(g), \text{в прямом направлении} \\ \varphi(g), \text{в обратном направлении} \end{cases}$$

Для рассмотренного примера:

$$\Delta\varphi = \min(4 - 3{,}3 - 1{,}3{,}2 - 0{,}2 - 0{,}4 - 2) = 1$$



Обобщенный алгоритм коррекции потока:

1. Производится разметка вершин сети, и если вершина стока достигается, то выполняется п.2, иначе делается вывод, что поток максимален.
2. Выделяется цепочка дуг, вычисляется величина $\Delta\varphi$ и производится коррекция потока на $\Delta\varphi$ в зависимости от направления прохода дуги от вершины истока к вершине стока по выделенной цепочке дуг.
3. Индексы всех вершин стираются и снова выполняется п.1 алгоритма.

В рассмотренном примере оказывается помеченной только вершина истока, а дальше разметка не проходит, т.к. все дуги, исходящие из этой вершины, насыщены, а дуг, входящих в нее, нет, что и означает максимальность скорректированного потока.

С тем, чтобы не перечерчивать многократно сеть, следует вести запись преобразований и процесса решения в форме таблицы (данные представлены для рассмотренного примера):

Номер преобр.	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	$\Delta\varphi$
I	+0	-5_3	$+1_1$		$+3_2$		$+2_2$			$+7_2$	$+10_2$	1
II	+0											

3.4.7.3 Задача о минимальном потоке

Задача построения минимального потока требует удовлетворения противоположного соотношения между потоком и функцией $C(g)$:

$$\forall g \in G(\varphi(g) \geq C(g)) \qquad (5.7)$$

Для этой задачи функцию $C(g)$ можно назвать **потребностью дуги**, и поток должен ее удовлетворить. Естественно, что в такой постановке нужно найти **минимальный возможный** поток, т.е.

$$V = \sum_{x_i \in G^{-1}(x_{st})} \varphi(x_i, x_{st}) \to min \qquad (5.8)$$

Построение произвольного потока, удовлетворяющего (5.7):

I. В исходной сети находится дуга g, для которой $\varphi(g) < C(g)$, и выделяется произвольный путь из вершины x_{is} к вершине x_{st}, содержащий эту дугу. Добавляется по дугам этого пути такой поток $\Delta\varphi$, чтобы для всех дуг пути выполнить условие (5.7):

$$\Delta\varphi = \max_{g \in (x_{is} \to x_{st})} (C(g) - \varphi(g)) \qquad (5.9)$$

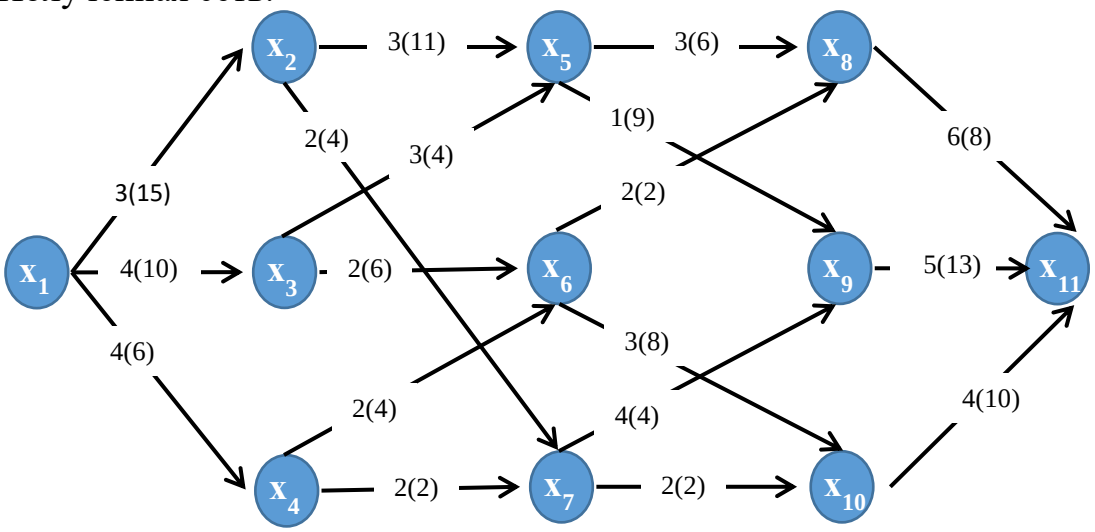
II. Если в сети есть еще дуги, для которых не выполнено условие (5.7), то снова выполняется п.I, иначе построение потока завершается

Пример: Построение произвольного потока.

Краткая запись выделяемых путей и необходимых увеличений потоков по каждой дуге сводится в таблицу, аналогичную той, что строилась для поиска максимального потока:

Номер преобр.	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	$\Delta\varphi$
I.	0	$+1_3$			$+2_3$			$+5_3$			$+8_6$	6
II.	0	$+1_0$			$+2_0$				$+5_1$		$+9_5$	5
III.	0	$+1_0$					$+2_2$		$+7_4$		$+9_0$	4
IV.	0		$+1_4$		$+3_3$				$+5_0$		$+9_0$	4
V.	0		$+1_0$			$+3_2$		$+6_2$			$+8_0$	2
VI.	0		$+1_0$			$+3_0$				$+6_3$	$+10_4$	4
VII.	0			$+1_4$		$+4_2$				$+6_0$	$+10_0$	4
VIII.	0			$+1_0$			$+4_2$			$+7_2$	$+10_0$	2

Полученная сеть:

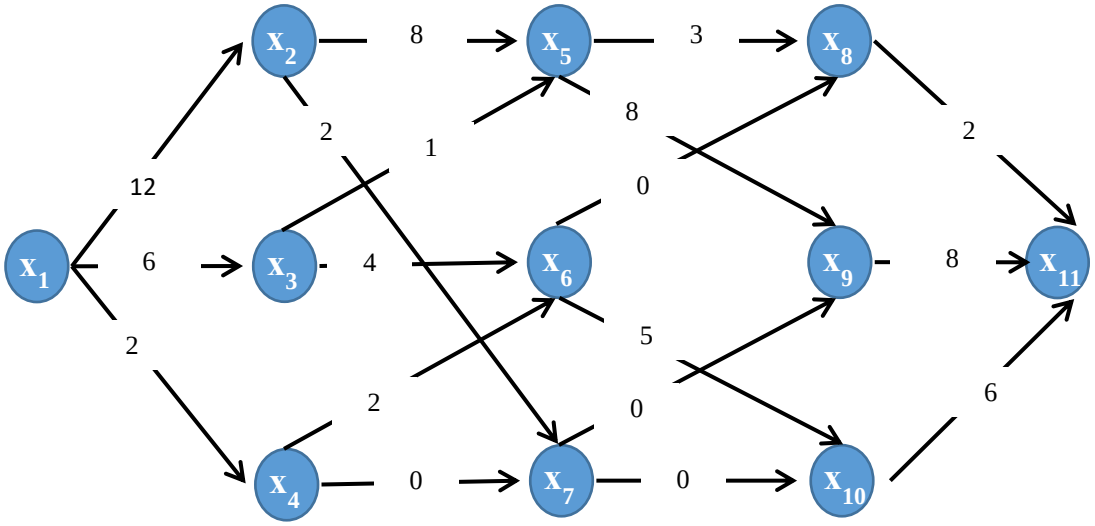


Произвольный поток φ^n, удовлетворяющий условию (5.7), построен, но по многим дугам протекает лишний поток.

Для его сокращения строится исходная сеть с пропускными способностями $C^l(g) = \varphi^n(g) - C(g)$. В рамках этих пропускных способностей $C^l(g)$ строится максимальный лишний поток $\varphi^l(g) \leq C^l(g)$.

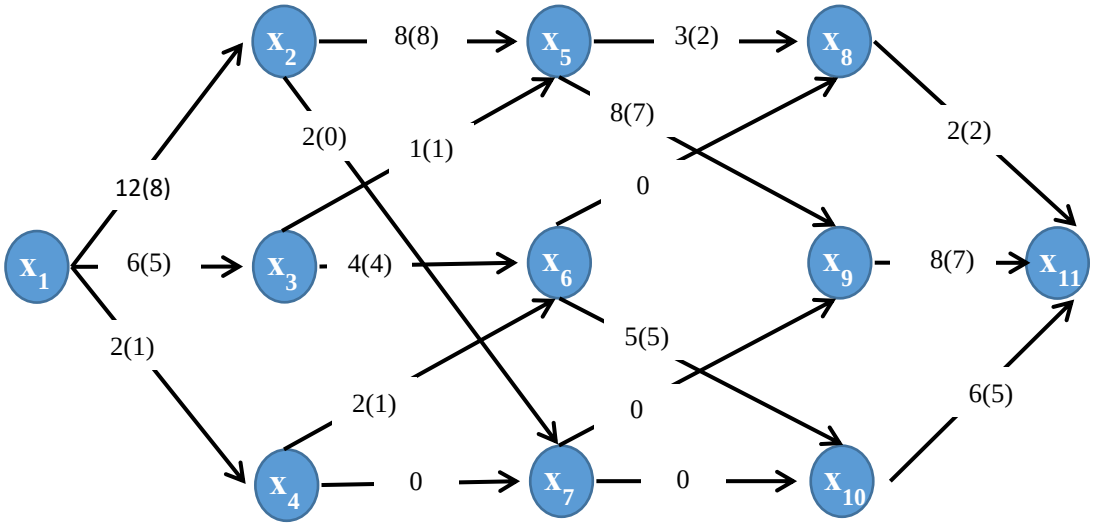
Пример:

Исходная сеть с пропускными способностями $C^l(g)$:



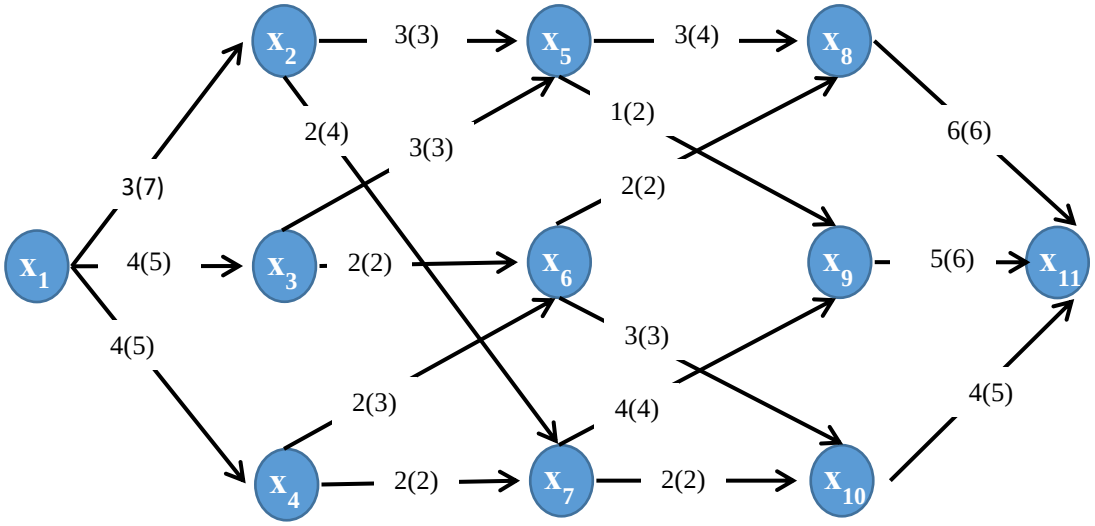
Построение максимального лишнего потока:

Номер преобр.	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	$\Delta\varphi$
I.	0	$+1_{12}$			$+2_8$			$+5_3$			$+8_2$	2
II.	0	$+1_{10}$			$+2_6$				$+5_6$		$+9_6$	6
III.	0		$+1_6$		$+3_1$				$+5_2$		$+9_2$	1
IV.	0		$+1_5$			$+3_4$				$+6_5$	$+10_6$	4
V.	0			$+1_2$		$+4_2$				$+6_1$	$+10_2$	1
VI.	0	$+1_4$	$+1_1$	$+1_1$		$+4_1$	$+2_2$					



Если из произвольного потока $\varphi^n(g)$ вычесть максимальный лишний поток $\varphi^l(g)$, то получится поток $\varphi(g) = \varphi^n(g) - \varphi^l(g)$, который будет минимальным по построению.

Пример: Построение минимального потока



Алгоритм построения минимального потока:

1. Строится произвольный поток $\varphi^n(g)$, удовлетворяющий условию (5.7).
2. В исходной сети формируется пропускная способность $C^l(g) = \varphi^n(g) - C(g)$ и в рамках этой пропускной способности строится максимальный лишний поток $\varphi^l(g)$.
3. Из произвольного потока вычитается максимальный лишний поток, получается искомый минимальный поток $\varphi(g) = \varphi^n(g) - \varphi^l(g)$.

3.4.8 Планарные графы

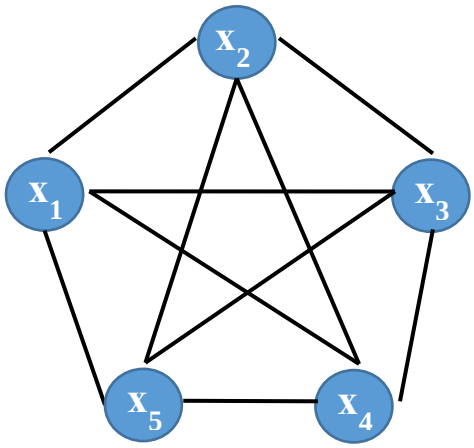
3.4.8.1 Определения и свойства планарного графа

Граф $G = (X, A)$ – **планарный**, если он может быть изображен на плоскости так, что не будет пересекающихся дуг.

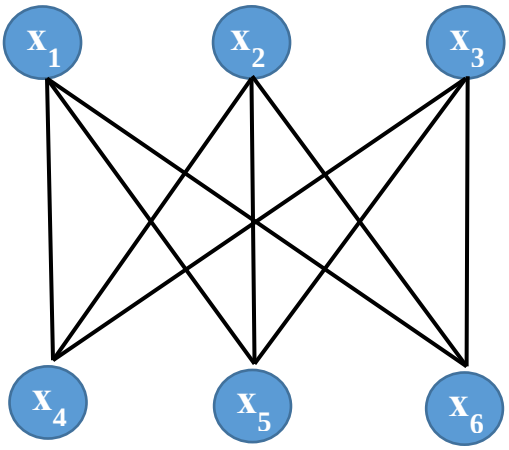
Для большинства графов нельзя построить плоское изображение. Такие графы не являются планарными.

Классические примеры непланарных графов:

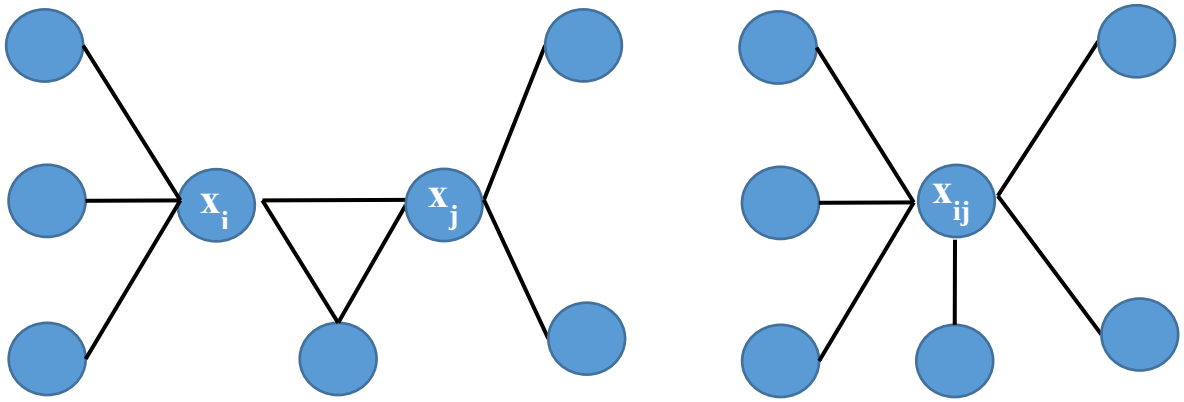
Полный пятивершник



«Три дома и три колодца»



Операция стягивания вершин – соединение связанных ребром вершин в одну с сохранением всех остальных связей:



Теорема. Граф не является планарным, если он содержит подграфы:

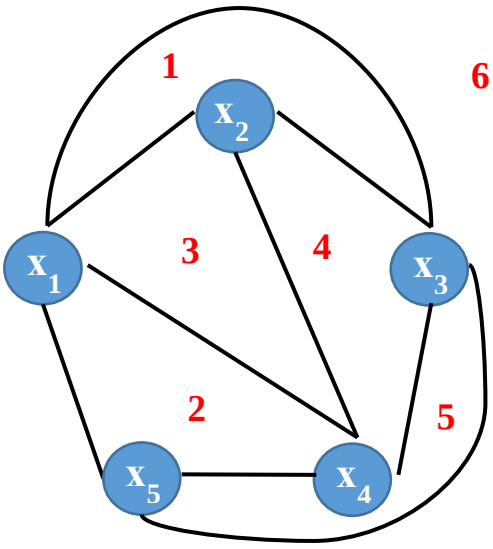
– полученные стягиванием вершин,
– вида полного пятивершника,
– вида «три дома и три колодца».

Однако данная теорема не дает алгоритма проверки планарности графа.

Плоское изображение графа производит разбиение плоскости на несколько отдельных фрагментов – один бесконечный (внешний) фрагмент и несколько конечных (внутренних) фрагментов. Такие фрагменты называются **гранями**.

Цикломатическое число Υ равно числу конечных граней.

Пример:



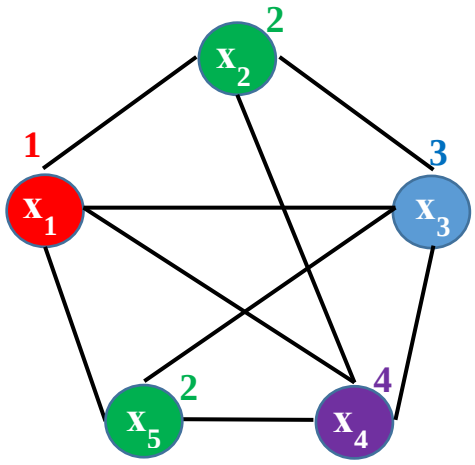
Получены 5 конечных граней (1-5) и одна бесконечная (6)

Цикломатическое число:
Υ=9-5+1=5

Теорема. Края конечных граней плоского изображения образуют систему независимых циклов графа.

Теорема. Для плоских (планарных) графов хроматическое число $\chi \leq 4$.

Пример:



Методом граничного перебора строим максимальные совместимые подмножества:

x_1	x_2	x_3	x_4	x_5	
1					1
	1			1	2
		1			3
			1		4

$\chi = 4$

Таким образом, 4 максимальных совместимых подмножеств входят в кратчайшее покрытие

Например, географические карты соответствуют плоскому изображению графа соседства стран (вершины – страны, ребра соединяют вершины, соответствующие странам с общей границей). Для окраски географических карт достаточно четырех красок.

3.4.8.2 Построение плоского изображения или выявление непланарности произвольного графа

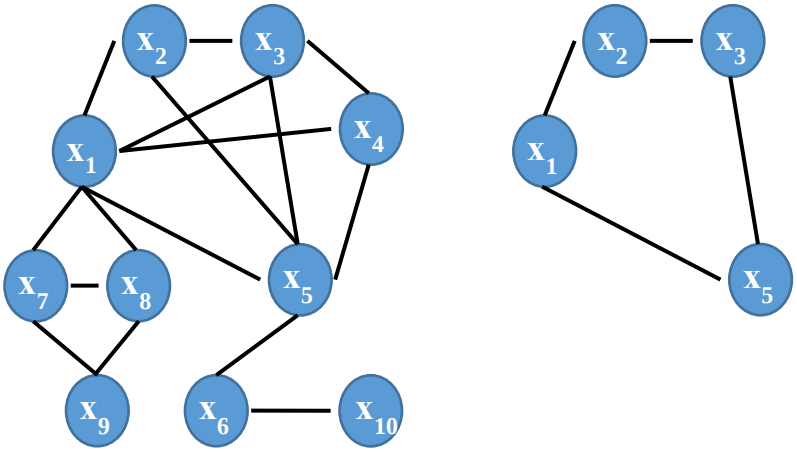
Для каждого графа необходимо либо построить его плоское изображение, либо определить, что он не планарный.

Если граф не содержит циклов, то он планарен. Построение плоского изображения дерева не представляет сложности

Прежде чем формулировать общий алгоритм введем некоторые понятия.

Если исходный граф не является деревом, то в нем можно выделить простой цикл с целью построения его плоского изображения.

Пример:



В исходном графе в общем случае остаются вершины и ребра, не вошедшие в полученное плоское изображение выделенного цикла.

Сегмент – частичный подграф исходного графа, обладающий одним из следующих свойств:

– ребро исходного графа, не вошедшее в плоское изображение, вершины которого принадлежат этому изображению;

(x_2, x_5), (x_1, x_3)

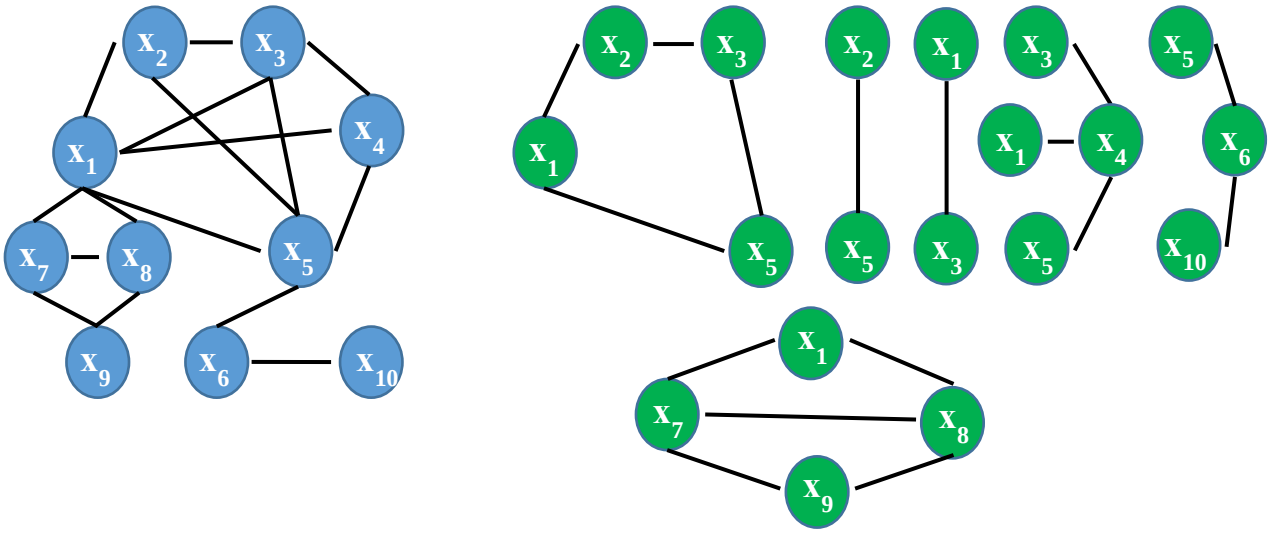
– связная компонента исходного графа, которая содержит одну или несколько вершин, не принадлежащих плоскому изображению, и все ребра, инцидентные этим вершинам

вершина x_4 *и ребра* (x_3, x_4), (x_1, x_4), (x_5, x_4);

вершины x_6, x_{10} *и ребра* (x_5, x_6), (x_6, x_{10});

вершины x_7,x_8,x_{10} *и ребра* (x_1, x_7), (x_1, x_8), (x_7, x_8), (x_7, x_9), (x_8, x_9).

Пример:



Вершины сегмента принадлежащие выделенному плоскому изображению, называются **контактными**.

Допустимой гранью сегмента назовем грань выделенного плоского подграфа, содержащую все контактные вершины сегмента.

Для рассмотренного примера 5.16 все сегменты имеют две допустимые грани (конечную и бесконечную).

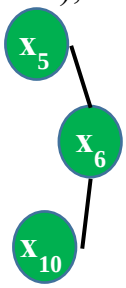
В общем случае различаются три ситуации:

1. для сегмента нет ни одной допустимой грани;
2. для сегмента существует одна допустимая грань;
3. для сегмента число допустимых граней больше одной.

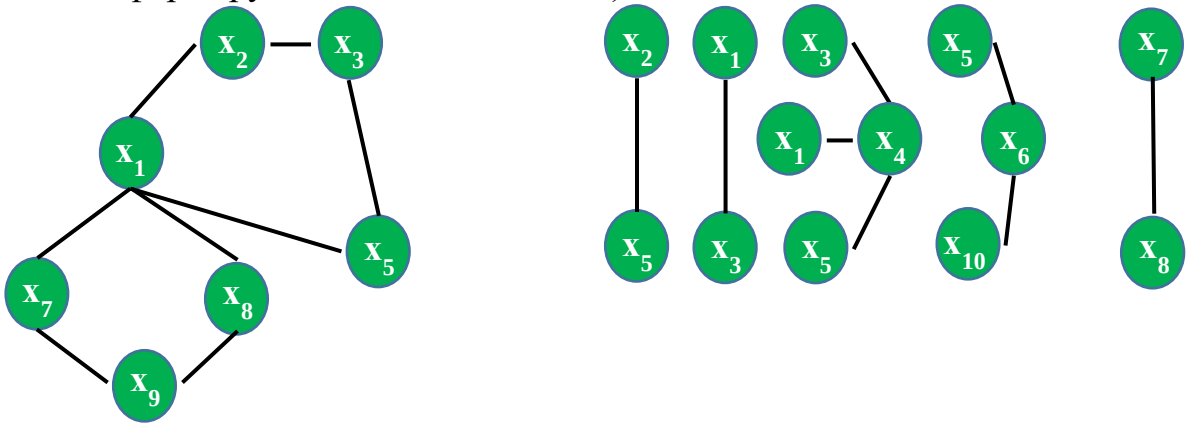
В любой допустимой грани *можно пытаться* строить плоское изображение сегмента, если оно существует. Однако, достаточно изобразить хотя бы его часть, чтобы получить новое выделенное плоское изображение и новые сегменты.

Рассмотрим три ситуации, когда сегмент имеет:

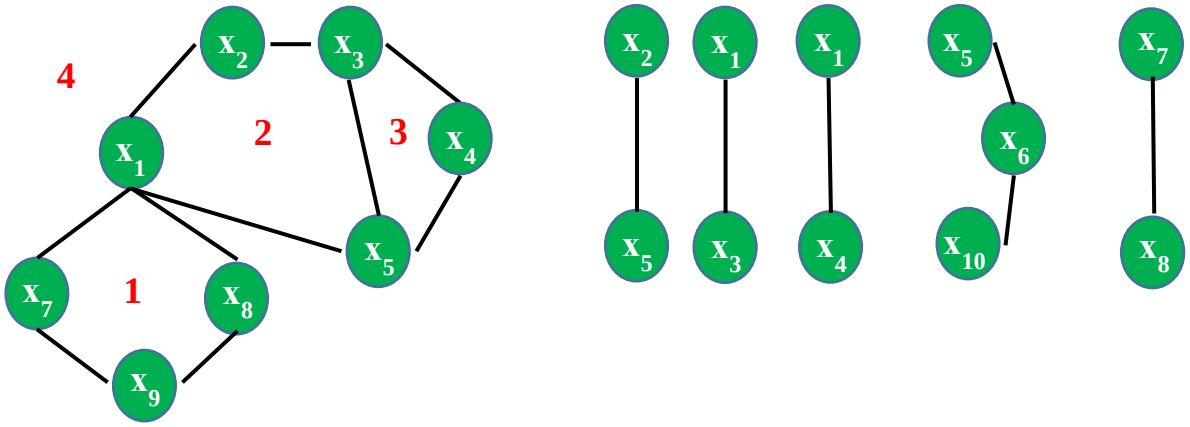
а) одну контактную вершину и представляет собой дерево, тогда можно построить полное его плоское изображение (при этом число граней не изменится);



б) одну контактную вершину, но не является деревом, тогда в нем можно выделить простой цикл к плоскому изображению (число граней увеличится и формируются новые сегменты);



в) более одной контактной вершины, тогда в нем можно выделить простой маршрут между этими вершинами и включить этот маршрут в плоское изображение (число граней увеличится и изменится множество сегментов)



Таким образом, построение плоского изображение планарного графа сводится к последовательному построению текущего плоского изображения частичного подграфа, выбору очередного сегмента и достройке плоского изображения в допустимой грани.

Алгоритм сводится к выбору очередного рассматриваемого сегмента и допустимой для него грани, возможны следующие три ситуации:

а) имеется сегмент, для которого нет ни одной допустимой грани, тогда построение плоского изображения невозможно и исходный граф непланарен;

б) имеется сегмент, для которого есть только одна допустимая грань, тогда существует единственный вариант помещения этого сегмента в этой грани;

в) для всех сегментов число допустимых граней больше одной, тогда можно организовать перебор вариантов, но для данной задачи можно обойтись и без него. В этой ситуации можно выбрать произвольный сегмент и поместить его в произвольную допустимую для него грань. При этом при любом выборе для планарного графа будет построено плоское изображение, а если на некотором шаге возникнет ситуация а), то вывод о непланарности графа остается справедливым.

Алгоритм

1. Если исходный граф – дерево, то строим его плоское изображение, иначе выбираем простой цикл, строим текущее плоское изображение и множество сегментов.

2. Сегменты, имеющие одну контактную вершину и являющиеся деревьями, присоединяем к плоскому изображению. Если множество сегментов пусто, то процесс построения плоского изображения завершен и граф планарен. Иначе если сегмент, для которого множество допустимых граней пусто, то граф непланарен. Иначе, если найдется сегмент с одной допустимой гранью, то выбираем его и его допустимую грань, выполняем п.3, иначе выбираем произвольный сегмент и произвольную его допустимую грань, выполняем п.3.

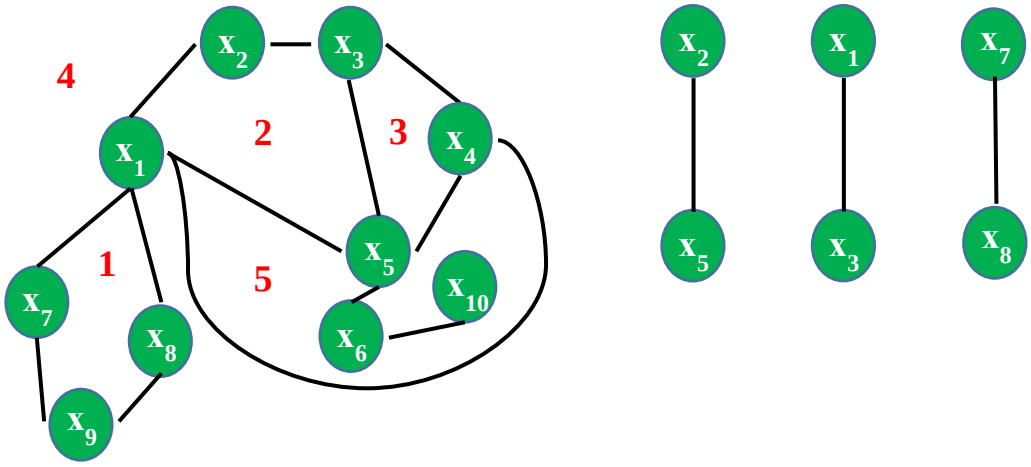
3. Если выбранный сегмент имеет одну контактную вершину, то выделяем в нем произвольный простой цикл и выполняем п.4. Если выбранный сегмент имеет две контактные вершины, то строим маршрут между этими вершинами и также выполняем п.4.

4. Дополняем выделенные циклы или маршрут в выбранную грань текущего плоского изображения, стоим новое множество сегментов. Выполняем п.2.

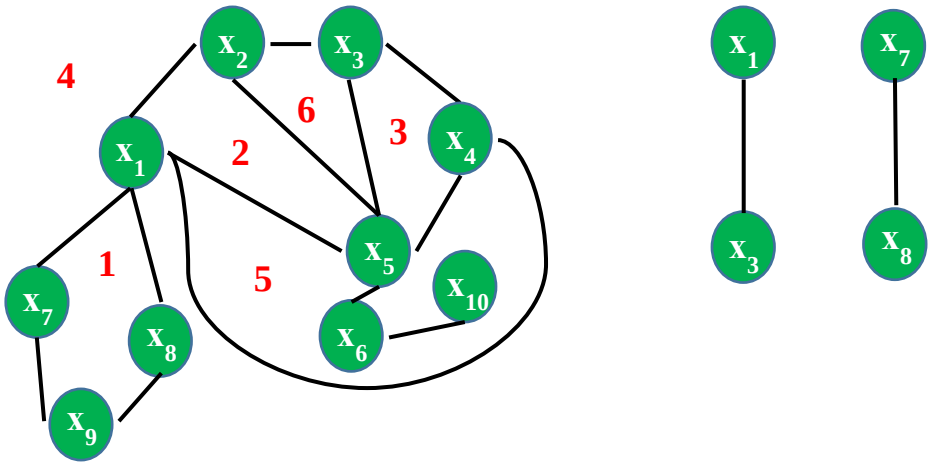
Пример:

Сегмент (x_5, x_6), (x_6, x_{10}) – дерево и может быть изображен произвольно в любой допустимой грани (4, 2, 3), выбираем грань 4.

Сегмент (x_1, x_4) имеет одну допустимую грань 4 и является маршрутом между контактными вершинами x_1 и x_4.

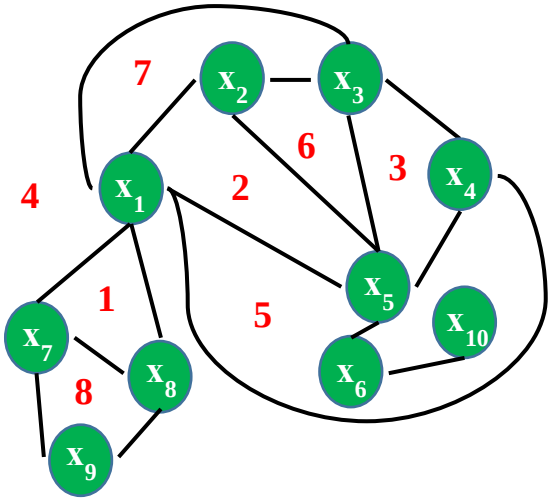


Сегмент (x_2, x_5) имеет одну допустимую грань - 2. Получаем новое плоское изображение и сегменты.



Сегмент (x_1, x_3) имеет одну допустимую грань – 4 и помещается в ней.

Сегмент (x_7, x_8) имеет одну допустимую грань – 1 и помещается в ней.



Получаем пустое множество сегментов и плоское изображение исходного графа.

ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ И ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Лабораторная работа № 1
ОПЕРАЦИИ НАД МНОЖЕСТВАМИ

Заданы множества ***A, B, C***. При $U = \{ u_i \mid u_i \in N \ \& \ u_i \le 20\}$, построить:

1) $X = A \cup B$,
2) $Y = A \cap B$,
3) $Z = A \backslash B$,
4) $\bar{C}$,
5) $D = Y \times Z$ (для полученных Y и Z),
6) множество всех подмножеств множества Y.

Индивидуальные задания по вариантам

1.
A={2,3,4,5,6,16}
B={5,6,20,15,11,1,16}
C={11,3,5,14,7,8,6,17}

2.
A={2,3,14,5,8,18}
B={5,18,20,15,11,1,2}
C={1,13,5,14,7,8,6,19}

3.
A={12,3,14,9,6,18}
B={5,6,3,15,11,1,16}
C={19,3,9,14,7,8,6,11}

4.
A={2,13,14,7,6,18}
B={6,2,15,13,18,3}
C={20,3,9,4,7,8,6,17}

5.
A={2,3,4,8,6,15}
B={5,6,20,10,11,1,15}
C={11,3,5,4,7,8,9,10}

6.
A={2,3,9,8,16,20}
B={5,6,2,20,11,1,16}
C={11,3,15,4,17,5,9,1}

7.
A={2,3,4,5,11,16}
B={5,6,1,11,9,16}
C={10,3,15,4,7,8,6,18}

8.
A={4,3,14,5,10,6}
B={5,6,1,10,9,3}
C={10,13,5,4,7,2,6,18}

9.
A={ 12,6,14,1,8,16 }
B={ 5,6,1,11,9,20 }
C={ 2,3,11,4,7,18,6,10 }

10.
A={12,6,11,7,10,19}
B={15,6,8,11,19,20}
C={5,9,10,4,7,8,6,20}

11.
A={16,6,10,1,2,8}
B={5,6,1,11,9,10}
C={2,3,11,4,8,17,6,12}

12.
A={16,6,10,1,2,8}
B={5,6,1,11,9,10}
C={11,4,8,2,3,17,6,12}

13.
A={5,6,14,1,8,16}
B={6,1,5,11,9,2 }
C={12,3,1,4,17,8,6,10}

14.
A={15,8,14,10,9,6}
B={6,1,8,11,9,2,20}
C={8,16,12,13,7,4,9,10}

15.
A={20,19,14,3,8,11}
B={5,6,3,11,9,20}
C={2,3,11,14,7, 8,16,10}

16.
A={12,1,14,13,8,2}
B={2,6,3,1,9,20}
C={13,11,4,7,18,2,16,10}

17.
A={12,3,14,11,5,1,13}
B={5,6,1,13,9,18}
C={12,3,10,4,7,18,6,1}

18.
A={10,14,12,8,15,1,13}
B={5,6,1,13,9,8}
C={12,9,10,4,17,5,6,1}

19.
A={4,6,14,7,8,19}
B={6,1,11,4,9,20}
C={2,3,10,4,11,18,6,13}

20.
A={12,7,14,6,8,9}
B={6,14,11,4,9,20}
C={12,15,1,4,20,18,6,13}

21.
A={7,16,14,2,8,19}
B={2,1,11,4,19,20}
C={12,20,4,11,18,6,3,13}

22.
A={12,15,4,6,10,9}
B={6,14,11,4,9,20}
C={20,15,1,4,3,18,6,13}

23.
A={14,6,17,3,18,2}
B={6,1,11,14,9,12}
C={2,3,20,14,1,18,6,13}

24.
A={15,7,14,6,18,19}
B={6,14,11,14,19,20}
C={2,15,1,4,20,8,16,3}

25.
A={14,5,14,17,8,9}
B={6,1,11,14,9,18}
C={2,15,16,11,4,20,8,13}

26.
A={16,8,13,7,19,20}
B={7,15,12,11,6,7,19}
C={2,3,19,14,1,18,6,13}

27.
A={12,15,4,6,10,9}
B={6,10,11,15,19,4}
C={12,15,1,4,20,18,6,13}

28.
A={20,9,14,2,8,11}
B={6,2,12,4,9,20}
C={13,11,4,7,18,2,6,20}

29.
A={15,5,9,1,3,7}
B={5,6,1,11,9,10}
C={2,3,11,4,8,17,6,12}

30.
A={16,6,10,1,2,8}
B={5,6,1,11,9,10}
C={11,4,8,2,3,17,6,12}

Лабораторная работа № 2
ОПЕРАЦИИ НАД МНОЖЕСТВАМИ

Используя основные аксиомы и теоремы теории множеств, упростить выражение. Задание выполнять пошагово с ОБЯЗАТЕЛЬНЫМ указанием соответствующего закона/аксиомы.

Индивидуальные задания по вариантам

$\overline{C \cup B \cup D} \cup \overline{B \cup \bar{D} \cup A \cap D} \cup \overline{C \cup B} \cap \bar{D}$;

$\overline{\overline{A \cup B \cup C} \cup \bar{A} \cup B} \cup A \cap C \cup \overline{\bar{A} \cup \bar{C}} \cap B$;

$\overline{\bar{A} \cup \bar{B}} \cup A \cup C \cup \bar{A} \cap C \cup \overline{\bar{A} \cup \bar{B} \cup \bar{C}} \cup B \cap C$;

$\overline{A \cup \bar{B} \cup C} \cup \overline{A \cup B \cap C} \cup \overline{\overline{A \cap C} \cup \bar{A} \cap B}$;

$\overline{\bar{C} \cup \bar{B} \cup \overline{\bar{B} \cup C}} \cup A \cup B \cup C \cup \bar{A} \cap D \cup \overline{A \cup B \cup D}$;

$\overline{\overline{A \cup B} \cap \overline{C \cap D}} \cup A \cap \overline{C \cup B} \cup A \cap \overline{B \cap D}$;

$\overline{A \cap B \cup D} \cup A \cap B \cap C \cup A \cap \bar{B} \cup \overline{\bar{B} \cup C}$;

$\overline{A \cap B} \cap C \cup \overline{\overline{A \cap \bar{B} \cap C} \cup A \cup \overline{B \cap C}} \cap \bar{C}$;

$\bar{A} \cap (\overline{(B \cup C) \cap (B \cup \bar{D})} \cup \overline{\bar{B} \cup \bar{D}}) \cup \bar{A} \cap \bar{D} \cup A \cap C$;

$\overline{(A \cup B \cup D) \cap C \cup \overline{A \cap \bar{C}}} \cup (\overline{C \cup A}) \cap B$;

$\overline{\overline{\bar{A} \cup \bar{B} \cap C} \cup \overline{A \cap B \cap C}} \cup D \cup A \cap B \cup C$;

$A \cap (\overline{\bar{A} \cup B \cup C} \cup (\bar{A} \cup \bar{B} \cup \bar{C}) \cap A \cap C) \cup A \cap \bar{C} \cup \bar{A} \cap D$;

$\overline{\overline{C \cup B \cup D} \cup \overline{B \cap \bar{D} \cup A \cap D}} \cup \overline{C \cup B} \cap \bar{D}$;

$\overline{A \cup B \cup \bar{C} \cup \overline{\bar{A} \cup \bar{B} \cup \overline{A \cap C}} \cup \overline{\bar{A} \cup \bar{C}} \cap B} \cup C$;

$\overline{\bar{A} \cup \bar{B}} \cup A \cap C \cup \bar{A} \cap C \cup \overline{\bar{A} \cup \bar{B} \cup C} \cup B \cap C$;

$\overline{A \cup \bar{B} \cup C} \cup \overline{\overline{A \cup B \cap C} \cup \overline{\overline{A \cap C} \cup \bar{A} \cap B}}$;

$\overline{\bar{B} \cup \bar{C} \cup \overline{\bar{B} \cup C}} \cup D \cap B \cup C \cap \bar{A} \cup \overline{A \cup B \cap D}$;

$\overline{\overline{B \cap A} \cap \overline{D \cap C}} \cup \overline{A \cap \overline{C \cup B}} \cup A \cap \overline{B \cap D}$;

$\overline{A \cap B \cup D} \cup A \cap \overline{B \cap C} \cup A \cap \bar{B} \cup \overline{\bar{D} \cup C}$;

$\overline{\overline{B \cap A} \cap C \cup \overline{A \cap \bar{B} \cap C}} \cup \overline{A \cup \overline{B \cap \bar{C}} \cap C}$;

$\overline{\overline{C \cup B \cap D} \cup \overline{B \cap \bar{D} \cup A \cap D}} \cup \overline{C \cap B} \cap \bar{D}$;

$\overline{C \cup B \cap D} \cup \overline{D \cap B \cup \bar{A} \cap D} \cup \overline{\overline{C \cap \bar{B}} \cap \bar{D}}$;

$\overline{\overline{A \cup B \cup C} \cup \overline{\bar{A} \cup B}} \cup A \cap C \cup \overline{A \cup \bar{C}} \cap B$;

$\overline{\bar{A} \cup \bar{B}} \cup A \cup C \cup A \cap C \cup \overline{\bar{A} \cup \bar{B} \cap C} \cup \bar{B} \cap C$;

$\overline{A \cup \bar{B} \cup C} \cup \overline{A \cap B \cup C} \cup \overline{\overline{A \cup C} \cap \bar{A} \cap B}$;

$(\overline{\bar{A} \cup B \cup C} \cup (\bar{A} \cup \bar{B} \cup \bar{C}) \cap \bar{A} \cup C) \cap \overline{\bar{C} \cup \bar{A} \cap D}$;

$\overline{A \cap \bar{B} \cup C} \cup \overline{\bar{A} \cup B} \cup \bar{A} \cap C \cup \overline{\bar{A} \cap \bar{C}} \cap B$;

$\overline{A \cap \bar{B}} \cup \bar{C} \cup \overline{\bar{A} \cup B \cap C} \cup \overline{A \cup \bar{C}} \cup \bar{A} \cap B$;

$\overline{\overline{A \cup B} \cap \overline{\bar{C} \cup D}} \cup \bar{A} \cap \overline{\bar{C} \cup B} \cup A \cap \overline{B \cap D}$;

$\overline{C \cap \overline{\bar{B} \cup D}} \cup \overline{B \cup \overline{\bar{D} \cup A \cap D}} \cup \overline{\bar{C} \cup B} \cup D$.

Лабораторная работа № 3
ОСНОВНЫЕ ОПЕРАЦИИ НАД ОТНОШЕНИЯМИ

Заданы отношения R_1 и R_2. Построить:

1) $R_1 \cup R_3$,
2) $R_1 \cap R_3$,
3) $\overline{R_1}$,
4) $R_1 \circ R_2$,
5) R^{-1}
6) $(R_1 \circ R_2)^{-1}$

Индивидуальные задания по вариантам

1.

R_1	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1		
x_3	1		1	

R_2	z_1	z_2	z_3
y_1	1		
y_2		1	
y_3	1	1	1
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1	1	1		
x_2			1	
x_3	1			

2.

R_1	y_1	y_2	y_3	y_4
x_1	1			
x_2		1		1
x_3			1	1

R_2	z_1	z_2	z_3
y_1		1	
y_2	1		1
y_3	1	1	1
y_4	1		

R_3	y_1	y_2	y_3	y_4
x_1		1	1	
x_2	1	1		
x_3				1

3.

R_1	y_1	y_2	y_3	y_4
x_1	1			
x_2	1	1		1
x_3	1		1	

R_2	z_1	z_2	z_3
y_1	1		1
y_2		1	
y_3	1	1	
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1	1	1		
x_2		1	1	1
x_3	1		1	

4.

R_1	y_1	y_2	y_3	y_4
x_1	1			
x_2		1		1
x_3			1	1

R_2	z_1	z_2	z_3
y_1		1	
y_2	1		
y_3	1	1	1
y_4	1		1

R_3	y_1	y_2	y_3	y_4
x_1		1		
x_2	1	1	1	
x_3			1	1

5.

R_1	y_1	y_2	y_3	y_4
x_1		1		
x_2	1	1		1
x_3		1	1	

R_2	z_1	z_2	z_3
y_1	1		1
y_2	1	1	
y_3	1	1	
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1		1		
x_2	1	1	1	
x_3	1			1

6.

R_1	y_1	y_2	y_3	y_4
x_1			1	1
x_2		1		1
x_3		1	1	1

R_2	z_1	z_2	z_3
y_1	1		1
y_2		1	
y_3	1	1	
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1	1			
x_2	1	1		
x_3	1		1	

7.

R_1	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	1
x_3		1	1	

R_2	z_1	z_2	z_3
y_1	1		1
y_2	1	1	
y_3	1	1	
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1			1	1
x_2	1	1	1	
x_3				

8.

R_1	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	1
x_3		1	1	

R_2	z_1	z_2	z_3
y_1		1	
y_2	1	1	
y_3	1	1	
y_4		1	1

R_3	y_1	y_2	y_3	y_4
x_1	1			1
x_2	1		1	
x_3		1		1

9.

R_1	y_1	y_2	y_3	y_4
x_1	1	1		
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1			1
y_2	1		1
y_3	1		1
y_4	1	1	

R_3	y_1	y_2	y_3	y_4
x_1	1			
x_2		1		1
x_3			1	

10.

R_1	y_1	y_2	y_3	y_4
x_1	1		1	
x_2		1	1	1
x_3		1	1	

R_2	z_1	z_2	z_3
y_1		1	
y_2	1	1	1
y_3			1
y_4	1	1	

R_3	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1		
x_3			1	

11.

R_1	y_1	y_2	y_3	y_4
x_1	1		1	1
x_2	1	1		
x_3		1		1

R_2	z_1	z_2	z_3
y_1			1
y_2	1	1	
y_3	1		1
y_4	1		1

R_3	y_1	y_2	y_3	y_4
x_1	1	1		
x_2		1	1	1
x_3				1

12.

R_1	y_1	y_2	y_3	y_4
x_1	1	1		
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1			1
y_2	1		1
y_3	1		
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	
x_3	1			

13.

R_1	y_1	y_2	y_3	y_4
x_1	1			1
x_2	1		1	1
x_3	1			1

R_2	z_1	z_2	z_3
y_1			1
y_2	1	1	1
y_3	1	1	
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1			1	1
x_2	1			
x_3		1		1

14.

R_1	y_1	y_2	y_3	y_4
x_1	1	1		
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1			1
y_2		1	
y_3	1	1	1
y_4	1		1

R_3	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	1
x_3			1	

15.

R_1	y_1	y_2	y_3	y_4
x_1	1		1	
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1			1
y_2	1	1	
y_3	1		1
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1		1		1
x_2	1	1		
x_3			1	1

16.

R_1	y_1	y_2	y_3	y_4
x_1		1	1	
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1		1	1
y_2	1		
y_3	1		1
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1	1		1
x_2	1		1	
x_3			1	

17.

R_1	y_1	y_2	y_3	y_4
x_1		1	1	
x_2		1	1	1
x_3		1		1

R_2	z_1	z_2	z_3
y_1		1	1
y_2	1		
y_3	1		1
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1	1		
x_2			1	
x_3		1		1

18.

R_1	y_1	y_2	y_3	y_4
x_1	1		1	
x_2	1	1		1
x_3		1		1

R_2	z_1	z_2	z_3
y_1	1	1	1
y_2			
y_3	1	1	1
y_4	1		1

R_3	y_1	y_2	y_3	y_4
x_1		1		1
x_2	1			1
x_3	1		1	

19.

R_1	y_1	y_2	y_3	y_4
x_1			1	
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1			1
y_2	1	1	
y_3	1		1
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1		1	
x_2		1		1
x_3	1	1		1

20.

R_1	y_1	y_2	y_3	y_4
x_1	1		1	1
x_2	1	1		
x_3		1		1

R_2	z_1	z_2	z_3
y_1			
y_2	1	1	
y_3			1
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1			
x_2		1	1	1
x_3		1		

21.

R_1	y_1	y_2	y_3	y_4
x_1	1	1	1	
x_2		1		
x_3		1		1

R_2	z_1	z_2	z_3
y_1		1	1
y_2	1	1	
y_3	1		
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1		1	
x_2	1	1		1
x_3				1

22.

R_1	y_1	y_2	y_3	y_4
x_1		1		1
x_2	1	1	1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1	1		1
y_2		1	
y_3	1		1
y_4		1	1

R_3	y_1	y_2	y_3	y_4
x_1	1			
x_2	1		1	
x_3		1	1	1

23.

R_1	y_1	y_2	y_3	y_4
x_1		1	1	
x_2	1		1	
x_3		1		1

R_2	z_1	z_2	z_3
y_1		1	1
y_2	1		
y_3			1
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	
x_3		1		

24.

R_1	y_1	y_2	y_3	y_4
x_1		1	1	1
x_2	1		1	
x_3			1	1

R_2	z_1	z_2	z_3
y_1	1		1
y_2	1	1	1
y_3			
y_4	1	1	

R_3	y_1	y_2	y_3	y_4
x_1	1		1	
x_2			1	
x_3	1	1		1

25.

R_1	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	
x_3	1		1	1

R_2	z_1	z_2	z_3
y_1	1		
y_2	1	1	1
y_3			1
y_4	1		1

R_3	y_1	y_2	y_3	y_4
x_1		1		
x_2			1	
x_3	1			1

26.

R_1	y_1	y_2	y_3	y_4
x_1	1	1		1
x_2		1		
x_3	1			1

R_2	z_1	z_2	z_3
y_1	1		
y_2		1	1
y_3	1	1	1
y_4			

R_3	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	
x_3	1			

27.

R_1	y_1	y_2	y_3	y_4
x_1		1	1	
x_2	1			1
x_3		1		1

R_2	z_1	z_2	z_3
y_1	1	1	1
y_2		1	
y_3	1		1
y_4		1	1

R_3	y_1	y_2	y_3	y_4
x_1	1	1	1	1
x_2				
x_3		1		

28.

R_1	y_1	y_2	y_3	y_4
x_1	1			1
x_2		1	1	
x_3	1		1	

R_2	z_1	z_2	z_3
y_1		1	1
y_2	1	1	
y_3		1	
y_4	1	1	1

R_3	y_1	y_2	y_3	y_4
x_1				1
x_2	1	1	1	
x_3				

29.

R_1	y_1	y_2	y_3	y_4
x_1	1		1	
x_2		1		1
x_3	1		1	

R_2	z_1	z_2	z_3
y_1	1		1
y_2	1	1	
y_3		1	1
y_4			1

R_3	y_1	y_2	y_3	y_4
x_1			1	
x_2		1		
x_3	1	1	1	1

30.

R_1	y_1	y_2	y_3	y_4
x_1		1		1
x_2		1	1	1
x_3	1		1	

R_2	z_1	z_2	z_3
y_1	1		
y_2	1	1	1
y_3		1	
y_4		1	1

R_3	y_1	y_2	y_3	y_4
x_1		1		1
x_2	1	1	1	1
x_3	1		1	

Лабораторная работа № 4
ПОИСК КРАТЧАЙШЕГО ПОКРЫТИЯ МЕТОДОМ РАЗЛОЖЕНИЯ ПО СТОЛБЦУ

Задана таблица покрытий. Построить хотя бы одно кратчайшее покрытие методом разложения по столбцу.

Индивидуальные задания по вариантам

1.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1							1
B	1	1					1		1	
C	1			1		1				1
D			1	1	1					
E						1			1	1
F							1	1		
G		1	1		1					
H						1		1		

2.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1					1	1		
B	1			1						
C						1			1	
D		1	1						1	1
E				1				1		
F	1					1				1
G			1		1				1	
H	1				1	1	1			

3.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1	1						1		
B		1	1		1		1			
C				1			1			
D				1				1		
E		1				1			1	1
F			1				1			1
G	1				1	1			1	
H			1			1				

4.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1		1	1	1		
B	1		1		1				1	
C				1	1	1				
D		1								1
E	1			1						
F		1	1				1			
G	1							1	1	
H						1				1

5.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1	1		1	1		
B			1	1		1				
C		1							1	
D					1			1		1
E			1					1		
F						1			1	
G	1		1			1	1			
H	1	1								1

6.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1						1		
B			1			1				
C	1	1			1					
D				1				1	1	
E	1			1				1		1
F				1		1				
G		1			1		1			1
H			1				1		1	

7.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1						1	1		1
B			1	1						
C					1			1	1	
D						1				1
E		1	1		1	1				
F	1	1								
G				1					1	
H			1			1	1			

8.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1			1				1		
B	1		1			1			1	
C					1			1		
D		1				1	1			
E					1				1	
F		1		1						
G			1	1				1		1
H		1					1			1

9.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1				1					
B		1		1		1				
C	1	1						1	1	
D			1		1		1			
E			1		1				1	
F	1					1	1	1		
G				1						1
H			1							1

10.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1	1						
B		1		1		1	1			
C					1	1				
D		1	1					1	1	
E	1				1					
F				1		1				1
G	1		1				1	1		
H	1								1	1

11.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1							1		1
B		1			1					
C				1			1			
D		1		1		1				1
E			1				1		1	
F	1		1			1	1			
G					1			1		
H		1		1					1	

12.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1					1	
B		1					1			1
C	1		1		1					1
D		1				1	1			
E			1						1	
F					1		1			
G	1			1				1		
H			1		1	1		1		

13.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1	1		1	1			
B		1						1	1	
C	1					1				1
D	1	1							1	
E				1			1	1		1
F				1	1					
G							1		1	
H			1		1					

14.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1				1			
B	1			1						
C			1	1		1			1	
D					1		1			
E	1						1	1		
F		1			1					
G				1				1	1	1
H		1				1				1

15.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1				1	1				
B		1		1						
C			1					1	1	1
D	1						1			
E				1					1	
F		1			1	1	1			1
G		1	1				1			
H	1				1			1		

16.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1		1				1	1
B		1		1				1		
C	1		1					1	1	
D		1					1			
E	1				1	1				
F		1		1						1
G						1	1			
H				1					1	

17.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1					1		1
B		1	1	1			1			
C				1		1	1			1
D	1					1			1	
E					1		1			
F	1			1						
G	1	1							1	
H					1		1			

18.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1	1	1			1				
B				1	1					1
C						1		1		
D		1			1	1			1	
E	1						1		1	
F		1		1						
G							1	1		
H			1	1					1	1

19.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1							1
B		1			1				1	
C			1	1	1					
D		1								1
E				1				1		
F		1			1	1	1			
G	1						1		1	1
H						1		1		

20.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1				1	1				1
B		1	1				1			
C								1	1	1
D			1		1	1		1		
E						1	1			
F				1					1	
G				1	1					
H	1	1					1			

21.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1			1				1	
B	1		1							
C					1		1			
D	1							1		
E		1	1	1			1			
F				1		1		1		1
G					1				1	1
H			1			1	1			

22.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1					1	1		1
B					1				1	
C	1		1	1						
D				1			1			
E			1	1				1		
F	1	1				1	1			
G			1		1					
H						1			1	1

23.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A						1			1	
B		1								1
C				1		1	1	1		
D			1				1		1	
E		1		1						
F				1	1	1				
G	1		1						1	
H	1				1			1		1

24.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1			1	1		1			
B			1				1			
C	1					1				1
D					1		1	1		
E		1		1				1	1	
F			1							1
G		1			1					
H		1				1			1	

25.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1	1			1		1
B				1			1			
C						1	1	1		
D	1		1	1						1
E		1							1	
F			1			1	1			
G	1	1			1					
H						1			1	

26.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1				1			1	
B	1		1							
C					1		1			
D	1							1		
E		1	1		1		1			
F				1		1		1		1
G					1				1	
H			1			1	1			1

27.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1						1	
B		1		1		1				
C	1	1					1	1	1	
D			1		1		1			
E			1		1				1	
F	1					1	1	1		
G				1						1
H			1			1				1

28.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1	1						
B			1	1		1		1		
C					1	1				
D		1	1				1	1	1	
E	1				1					
F		1		1		1				1
G	1		1				1	1		
H	1								1	1

29.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1			1	1				
B	1			1						
C			1					1	1	1
D		1					1			
E				1				1		
F	1				1	1	1			1
G	1		1				1			
H		1			1				1	

30.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1						1	1
B		1		1				1		
C			1		1			1		
D		1					1		1	
E	1				1	1				
F		1		1						1
G						1	1			
H				1					1	

Лабораторная работа № 5
ПОИСК МИНИМАЛЬНОГО ПОКРЫТИЯ МЕТОДОМ РАЗЛОЖЕНИЯ ПО СТОЛБЦУ

Задана таблица покрытий. Построить хотя бы одно минимальное покрытие методом разложения по столбцу.

Индивидуальные задания по вариантам

1.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1	1			1			**1**
B	1				1		1			**1**
C		1	1	1					1	**2**
D		1		1	1		1	1		**4**
E	1		1			1			1	**3**
F					1			1	1	**1**
G		1				1		1		**2**
H			1						1	**4**

2.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A				1	1		1			**1**
B	1					1			1	**2**
C	1			1	1			1	1	**3**
D			1					1	1	**1**
E		1	1			1	1			**3**
F	1	1		1		1	1			**2**
G		1			1			1		**2**
H	1			1						**4**

3.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1	1		1			1		**3**
B	1		1	1				1		**2**
C	1			1		1	1		1	**4**
D		1				1			1	**1**
E			1	1					1	**2**
F	1						1	1		**2**
G					1	1	1			**1**
H			1					1		**3**

4.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1		1				1		**2**
B			1			1			1	**1**
C		1			1		1		1	**3**
D	1			1			1		1	**2**
E	1		1		1					**1**
F	1		1	1		1		1		**3**
G			1						1	**2**
H						1	1	1		**2**

5.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1					1	1		**1**
B						1		1	1	**2**
C		1	1		1	1				**3**
D			1	1					1	**2**
E	1				1		1			**1**
F	1		1	1		1				**2**
G							1	1		**2**
H	1			1			1	1	1	**3**

6.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1	1					**2**
B	1			1		1	1		1	**3**
C					1		1		1	**1**
D		1				1	1			**2**
E		1	1		1			1		**3**
F	1		1	1	1					**2**
G	1					1		1		**2**
H			1	1					1	**1**

7.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1					1		1
B		1		1		1				2
C	1	1		1	1			1		3
D	1			1		1			1	2
E	1				1		1			1
F		1			1				1	2
G		1			1			1		4
H			1			1	1		1	3

8.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1				1		1		1
B		1		1			1		1	3
C				1	1			1	1	2
D			1			1			1	2
E	1				1		1			1
F	1		1	1						2
G			1			1				3
H	1		1		1	1		1		3

9.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1					1	2
B	1						1		1	1
C	1		1			1		1		3
D					1		1	1		2
E		1		1	1		1		1	4
F		1	1	1						1
G		1			1	1				1
H	1						1			3

10.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1	1						1		1
B		1		1	1		1	1		4
C		1		1		1				2
D					1	1	1			1
E			1				1	1		1
F	1		1			1			1	3
G		1						1		3
H				1	1				1	2

11.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1		1				1	1
B		1	1			1	1			3
C		1	1	1					1	2
D	1			1	1			1	1	3
E					1	1		1		1
F	1	1		1		1				2
G	1						1	1		2
H		1	1						1	3

12.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A					1		1	1		2
B	1			1		1				1
C	1	1	1							1
D		1				1	1		1	3
E			1		1		1		1	2
F	1		1	1	1			1		4
G				1				1	1	1
H	1			1						3

13.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1		1				1	1
B		1	1	1				1		3
C			1	1			1		1	2
D	1				1	1	1		1	4
E					1	1		1		2
F	1	1					1			1
G			1	1					1	4
H	1			1		1				1

14.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1	1					1			2
B				1	1				1	2
C					1	1	1	1		2
D		1				1			1	1
E	1		1		1	1				3
F			1	1				1		1
G		1		1			1	1	1	4
H		1				1				3

15.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1	1		1		1	4
B	1		1			1	1			2
C	1			1				1		1
D			1				1		1	1
E				1	1	1				2
F		1			1				1	1
G		1		1		1		1		3
H				1		2				3

16.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1			1	1				2
B	1	1		1						1
C	1	1			1			1	1	4
D			1					1	1	2
E	1					1	1	1		2
F					1		1		1	1
G		1		1						3
H			1	1		1	1			3

17.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1	1	1			1	1		3
B			1	1	1					1
C			1				1		1	2
D	1	1		1						1
E		1				1		1		1
F					1	1	1	1		2
G	1				1	1			1	3
H			1	1				1		3

18.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1			1	1	1			3
B		1						1	1	1
C	1		1				1			2
D			1		1	1			1	2
E	1		1	1				1	1	3
F	1			1		1				1
G		1						1		2
H				1	1			1		1

19.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1					1	2
B			1			1		1		1
C			1	1	1	1			1	3
D	1			1	1		1			2
E	1	1					1	1		3
F			1				1		1	1
G				1			1			3
H		1			1	1				1

20.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1	1			1			1
B		1	1		1				1	2
C					1	1		1		2
D	1			1	1				1	3
E							1	1	1	1
F	1	1				1				1
G		1	1			1	1	1		3
H							1	1		2

21.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1				1		1
B		1	1			1	1			2
C	1				1				1	1
D			1		1	1				2
E	1			1	1	1	1			3
F		1	1					1	1	3
G	1				1	1				3
H		1		1			1			1

22.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A				1			1	1	1	3
B		1	1						1	2
C	1	1	1		1	1				3
D			1	1	1			1		2
E	1					1	1			1
F	1			1	1					1
G			1					1		3
H		1				1		1		1

23.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A					1		1		1	1
B	1		1			1			1	3
C			1	1		1				2
D	1	1					1			1
E		1	1					1		2
F		1		1	1		1	1		3
G					1	1		1		1
H			1			1				3

24.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1			1			1	2
B	1				1		1	1		2
C		1	1	1						1
D	1			1	1				1	3
E					1	1		1		2
F	1	1					1			1
G		1	1			1	1	1		3
H		1					1			2

25.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1		1				1	1
B	1			1					1	2
C		1					1	1		1
D	1		1		1			1		2
E	1		1			1	1			3
F		1		1	1			1	1	3
G		1		1		1				2
H	1				1			1		3

26.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1	1					1			2
B				1	1				1	2
C				1		1	1	1		2
D		1				1			1	1
E	1		1	1		1				3
F			1		1			1		1
G		1			1		1	1	1	4
H		1				1				3

27.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1				1			1
B		1			1	1				2
C	1	1		1	1				1	3
D	1			1		1		1		2
E	1				1		1			1
F		1		1				1		2
G		1		1					1	4
H			1			1	1	1		3

28.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1				1		1		1
B		1		1			1		1	3
C				1	1			1	1	2
D			1			1			1	2
E	1				1		1			1
F	1		1	1						2
G			1			1				3
H	1		1		1	1		1		3

29.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1		1		1		1	1		4
B	1			1		1	1			2
C	1		1						1	1
D				1			1	1		1
E			1		1	1				2
F		1			1			1		1
G		1	1			1			1	3
H			1			2				3

30.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1			1	1				2
B	1	1	1							1
C	1	1			1			1	1	4
D				1				1	1	2
E	1					1	1	1		2
F					1	1			1	1
G		1	1							3
H			1	1		1	1			3

Лабораторная работа № 6
ПОИСК КРАТЧАЙШЕГО ПОКРЫТИЯ МЕТОДОМ ВЕТВЕЙ И ГРАНИЦ

Задана таблица покрытий. Построить хотя бы одно кратчайшее покрытие методом ветвей и границ.

Индивидуальные задания по вариантам

1.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1							1
B	1	1					1		1	
C	1			1		1				1
D			1	1	1					
E						1			1	1
F							1	1		
G		1	1		1					
H						1		1		

2.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1					1	1		
B	1			1						
C						1			1	
D		1	1						1	1
E				1				1		
F	1					1				1
G			1		1				1	
H	1				1	1	1			

3.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1	1						1		
B		1	1		1		1			
C				1			1			
D				1				1		
E		1				1			1	1
F			1				1			1
G	1				1	1			1	
H			1			1				

4.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1		1	1	1		
B	1		1		1				1	
C				1	1	1				
D		1								1
E	1			1						
F		1	1				1			
G	1							1	1	
H						1				1

5.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1	1		1	1		
B			1	1		1				
C		1							1	
D					1			1		1
E			1					1		
F						1			1	
G	1		1			1	1			
H	1	1								1

6.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1						1		
B			1			1				
C	1	1			1					
D				1				1	1	
E	1			1				1		1
F				1		1				
G		1			1		1			1
H			1				1		1	

7.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1						1	1		1
B			1	1						
C					1			1	1	
D						1				1
E		1	1		1	1				
F	1	1								
G				1					1	
H			1			1	1			

8.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1			1				1		
B	1		1			1			1	
C					1			1		
D		1				1	1			
E					1				1	
F		1		1						
G			1	1				1		1
H		1					1			1

9.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1				1					
B		1		1		1				
C	1	1						1	1	
D			1		1		1			
E			1		1				1	
F	1					1	1	1		
G				1						1
H			1							1

10.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1	1						
B		1		1		1	1			
C					1	1				
D		1	1					1	1	
E	1				1					
F				1		1				1
G	1		1				1	1		
H	1								1	1

11.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1							1		1
B		1			1					
C				1			1			
D		1		1		1				1
E			1				1		1	
F	1		1			1	1			
G					1			1		
H		1		1					1	

12.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1					1	
B		1					1			1
C	1		1		1					1
D		1				1	1			
E			1						1	
F					1		1			
G	1			1				1		
H			1		1	1		1		

13.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1	1		1	1			
B		1						1	1	
C	1					1				1
D	1	1							1	
E				1			1	1		1
F				1	1					
G							1		1	
H			1		1					

14.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1				1			
B	1			1						
C			1	1		1			1	
D					1		1			
E	1						1	1		
F		1			1					
G				1				1	1	1
H		1				1				1

15.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1				1	1				
B		1		1						
C			1					1	1	1
D	1						1			
E				1					1	
F		1			1	1	1			1
G		1	1				1			
H	1				1			1		

16.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1		1				1	1
B		1		1				1		
C	1		1					1	1	
D		1					1			
E	1				1	1				
F		1		1						1
G						1	1			
H				1					1	

17.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1					1		1
B		1	1	1			1			
C				1		1	1			1
D	1					1			1	
E					1		1			
F	1			1						
G	1	1							1	
H					1		1			

18.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1	1	1			1				
B				1	1					1
C						1		1		
D		1			1	1			1	
E	1						1		1	
F		1		1						
G							1	1		
H			1	1					1	1

19.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1							1
B		1			1				1	
C			1	1	1					
D		1								1
E				1				1		
F		1			1	1	1			
G	1						1		1	1
H						1		1		

20.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1				1	1				1
B		1	1				1			
C								1	1	1
D			1		1	1		1		
E						1	1			
F				1					1	
G				1	1					
H	1	1					1			

21.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1			1				1	
B	1		1							
C					1		1			
D	1							1		
E		1	1	1			1			
F				1		1		1		1
G					1				1	1
H			1			1	1			

22.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1					1	1		1
B					1				1	
C	1		1	1						
D				1			1			
E			1	1				1		
F	1	1				1	1			
G			1		1					
H						1			1	1

23.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A						1			1	
B		1								1
C				1		1	1	1		
D			1				1		1	
E		1		1						
F				1	1	1				
G	1		1						1	
H	1				1			1		1

24.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1			1	1		1			
B			1				1			
C	1					1				1
D					1		1	1		
E		1		1				1	1	
F			1							1
G		1			1					
H		1				1			1	

25.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A				1	1			1		1
B				1			1			
C						1	1	1		
D	1		1	1						1
E		1							1	
F			1			1	1			
G	1	1			1					
H						1			1	

26.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1				1			1	
B	1		1							
C					1		1			
D	1							1		
E		1	1		1		1			
F				1		1		1		1
G					1				1	
H			1			1	1			1

27.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1						1	
B		1		1		1				
C	1	1					1	1	1	
D			1		1		1			
E			1		1				1	
F	1					1	1	1		
G				1						1
H			1			1				1

28.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A			1	1						
B			1	1		1		1		
C					1	1				
D		1	1				1	1	1	
E	1				1					
F		1		1		1				1
G	1		1				1	1		
H	1								1	1

29.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A		1			1	1				
B	1			1						
C			1					1	1	1
D		1					1			
E				1				1		
F	1				1	1	1			1
G	1		1				1			
H		1			1				1	

30.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
A	1		1						1	1
B		1		1				1		
C			1		1			1		
D		1					1		1	
E	1				1	1				
F		1		1						1
G						1	1			
H				1					1	

Лабораторная работа № 7
ПОИСК МИНИМАЛЬНОГО ПОКРЫТИЯ МЕТОДОМ ВЕТВЕЙ И ГРАНИЦ

Задана таблица покрытий. Построить хотя бы одно минимальное покрытие методом ветвей и границ.

Индивидуальные задания по вариантам

1.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1	1			1			**1**
B	1				1		1			**1**
C		1	1	1					1	**2**
D		1		1	1		1	1		**4**
E	1		1			1			1	**3**
F					1			1	1	**1**
G		1				1		1		**2**
H			1						1	**4**

2.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A				1	1		1			**1**
B	1					1			1	**2**
C	1			1	1			1	1	**3**
D			1					1	1	**1**
E		1	1			1	1			**3**
F	1	1		1		1	1			**2**
G		1			1			1		**2**
H	1			1						**4**

3.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1	1		1			1		**3**
B	1		1	1				1		**2**
C	1			1		1	1		1	**4**
D		1				1			1	**1**
E			1	1					1	**2**
F	1						1	1		**2**
G					1	1	1			**1**
H			1					1		**3**

4.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1		1				1		**2**
B			1			1			1	**1**
C		1			1		1		1	**3**
D	1			1			1		1	**2**
E	1		1		1					**1**
F	1		1	1		1		1		**3**
G			1						1	**2**
H						1	1	1		**2**

5.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1					1	1		**1**
B						1		1	1	**2**
C		1	1		1	1				**3**
D			1	1					1	**2**
E	1				1		1			**1**
F	1		1	1		1				**2**
G							1	1		**2**
H	1			1			1	1	1	**3**

6.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1	1					**2**
B	1			1		1	1		1	**3**
C					1		1		1	**1**
D		1				1	1			**2**
E		1	1		1			1		**3**
F	1		1	1	1					**2**
G	1					1		1		**2**
H			1	1					1	**1**

7.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1					1		1
B		1		1		1				2
C	1	1		1	1			1		3
D	1			1		1			1	2
E	1				1		1			1
F		1			1				1	2
G		1			1			1		4
H			1			1	1		1	3

8.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1				1		1		1
B		1		1			1		1	3
C				1	1			1	1	2
D			1			1			1	2
E	1				1		1			1
F	1		1	1						2
G			1			1				3
H	1		1		1	1		1		3

9.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1					1	2
B	1						1		1	1
C	1		1			1		1		3
D					1		1	1		2
E		1		1	1		1		1	4
F		1	1	1						1
G		1			1	1				1
H	1						1			3

10.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1	1						1		1
B		1		1	1		1	1		4
C		1		1		1				2
D					1	1	1			1
E			1				1	1		1
F	1		1			1			1	3
G		1						1		3
H				1	1				1	2

11.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1		1				1	1
B		1	1			1	1			3
C		1	1	1					1	2
D	1			1	1			1	1	3
E					1	1		1		1
F	1	1		1		1				2
G	1						1	1		2
H		1	1						1	3

12.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A					1		1	1		2
B	1			1		1				1
C	1	1	1							1
D		1				1	1		1	3
E			1		1		1		1	2
F	1		1	1	1			1		4
G				1				1	1	1
H	1			1						3

13.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1		1				1	1
B		1	1	1				1		3
C			1	1			1		1	2
D	1				1	1	1		1	4
E					1	1		1		2
F	1	1					1			1
G			1	1					1	4
H	1			1		1				1

14.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1	1					1			2
B				1	1				1	2
C					1	1	1	1		2
D		1				1			1	1
E	1		1		1	1				3
F			1	1				1		1
G		1		1			1	1	1	4
H		1				1				3

15.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1	1		1		1	4
B	1		1			1	1			2
C	1			1				1		1
D			1				1		1	1
E				1	1	1				2
F		1			1				1	1
G		1		1		1		1		3
H				1		2				3

16.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1			1	1				2
B	1	1		1						1
C	1	1			1			1	1	4
D			1					1	1	2
E	1					1	1	1		2
F					1		1		1	1
G		1		1						3
H			1	1		1	1			3

17.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1	1	1			1	1		3
B			1	1	1					1
C			1				1		1	2
D	1	1		1						1
E		1				1		1		1
F					1	1	1	1		2
G	1				1	1			1	3
H			1	1				1		3

18.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1			1	1	1			3
B		1						1	1	1
C	1		1				1			2
D			1		1	1			1	2
E	1		1	1				1	1	3
F	1			1		1				1
G		1						1		2
H				1	1			1		1

19.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1					1	2
B			1			1		1		1
C			1	1	1	1			1	3
D	1			1	1		1			2
E	1	1					1	1		3
F			1				1		1	1
G				1			1			3
H		1			1	1				1

20.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1	1			1			1
B		1	1		1				1	2
C					1	1		1		2
D	1			1	1				1	3
E							1	1	1	1
F	1	1				1				1
G		1	1			1	1	1		3
H							1	1		2

21.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1			1				1		1
B		1	1			1	1			2
C	1				1				1	1
D			1		1	1				2
E	1			1	1	1	1			3
F		1	1					1	1	3
G	1				1	1				3
H		1		1			1			1

22.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A				1			1	1	1	3
B		1	1						1	2
C	1	1	1		1	1				3
D			1	1	1			1		2
E	1					1	1			1
F	1			1	1					1
G			1					1		3
H		1				1		1		1

23.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A					1		1		1	1
B	1		1			1			1	3
C			1	1		1				2
D	1	1					1			1
E		1	1					1		2
F		1		1	1		1	1		3
G					1	1		1		1
H			1			1				3

24.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1			1			1	2
B	1				1		1	1		2
C		1	1	1						1
D	1			1	1				1	3
E					1	1		1		2
F	1	1					1			1
G		1	1			1	1	1		3
H		1					1			2

25.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1		1				1	1
B	1			1					1	2
C		1					1	1		1
D	1		1		1			1		2
E	1		1			1	1			3
F		1		1	1			1	1	3
G		1		1		1				2
H	1				1			1		3

26.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1	1					1			2
B				1	1				1	2
C				1		1	1	1		2
D		1				1			1	1
E	1		1	1		1				3
F			1		1			1		1
G		1			1		1	1	1	4
H		1				1				3

27.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A			1				1			1
B		1			1	1				2
C	1	1		1	1				1	3
D	1			1		1		1		2
E	1				1		1			1
F		1		1				1		2
G		1		1					1	4
H			1			1	1	1		3

28.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1				1		1		1
B		1		1			1		1	3
C				1	1			1	1	2
D			1			1			1	2
E	1				1		1			1
F	1		1	1						2
G			1			1				3
H	1		1		1	1		1		3

29.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A	1		1		1		1	1		4
B	1			1		1	1			2
C	1		1						1	1
D				1			1	1		1
E			1		1	1				2
F		1			1			1		1
G		1	1			1			1	3
H			1			2				3

30.

R	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	c
A		1			1	1				2
B	1	1	1							1
C	1	1			1			1	1	4
D				1				1	1	2
E	1					1	1	1		2
F					1	1			1	1
G		1	1							3
H			1	1		1	1			3

Лабораторная работа № 8
МАКСИМАЛЬНЫЕ СОВМЕСТИМЫЕ ПОДМНОЖЕСТВА

На множестве ***X*** задано отношение несовместимости***R***.

Построить все максимальные совместимые подмножества множества ***X***.

Индивидуальные задания по вариантам

1.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1	1	
x_2	1	–	1			1
x_3		1	–	1		
x_4	1		1	–		1
x_5	1				–	1
x_6		1		1	1	–

2.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1			1
x_2		–	1	1		
x_3	1	1	–			1
x_4		1		–	1	1
x_5				1	–	1
x_6	1		1	1	1	–

3.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1			1	1
x_2	1	–		1		1
x_3			–	1	1	
x_4		1	1	–		1
x_5	1		1		–	
x_6	1	1		1		–

4.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1	1		
x_2		–	1		1	
x_3	1	1	–			1
x_4	1			–	1	
x_5		1		1	–	1
x_6			1		1	–

5.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1			1	
x_2	1	–		1	1	
x_3			–	1	1	1
x_4		1	1	–		
x_5	1	1	1		–	1
x_6			1		1	–

6.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1	1		1
x_2		–	1		1	
x_3	1	1	–		1	
x_4	1			–		1
x_5		1	1		–	1
x_6	1			1	1	–

7.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–			1	1	1
x_2		–	1		1	
x_3		1	–			1
x_4	1			–	1	1
x_5	1	1		1	–	
x_6	1		1	1		–

8.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1		1	1
x_2		–	1	1		
x_3	1	1	–			1
x_4		1		–	1	1
x_5	1			1	–	
x_6	1		1	1		–

9.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1			1	1
x_2	1	–		1		1
x_3			–	1	1	
x_4		1	1	–		
x_5	1		1		–	1
x_6	1	1			1	–

10.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1				1
x_2	1	–		1	1	
x_3			–	1	1	1
x_4		1	1	–		1
x_5	1	1			–	
x_6	1		1	1		–

11.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1	1		1
x_2		–	1		1	
x_3	1	1	–			1
x_4	1			–	1	
x_5		1		1	–	1
x_6	1		1		1	–

12.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1		1
x_2	1	–			1	1
x_3			–	1	1	
x_4	1		1	–		
x_5		1	1		–	1
x_6	1	1			1	–

13.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1		1
x_2	1	–	1			1
x_3		1	–	1	1	
x_4	1		1	–	1	
x_5			1	1	–	
x_6	1	1				–

14.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1		1	1
x_2		–	1	1	1	
x_3	1	1	–			1
x_4		1		–	1	
x_5	1	1		1	–	
x_6	1		1			–

15.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–			1	1	
x_2		–	1	1		
x_3		1	–		1	1
x_4	1	1		–		
x_5	1		1		–	1
x_6			1		1	–

16.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1			1	
x_2	1	–		1		1
x_3			–	1		1
x_4		1	1	–		
x_5	1				–	1
x_6		1	1		1	–

17.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1				1
x_2	1	–		1	1	
x_3			–		1	1
x_4		1		–	1	1
x_5		1	1	1	–	
x_6	1		1	1		–

18.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1			1
x_2		–	1	1		1
x_3	1	1	–		1	
x_4		1		–	1	1
x_5			1	1	–	
x_6	1	1		1		–

19.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1		1	1
x_2		–	1	1		1
x_3	1	1	–	1		
x_4		1	1	–		1
x_5	1				–	
x_6	1	1		1		–

20.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1	1	
x_2	1	–			1	
x_3			–	1		1
x_4	1		1	–	1	1
x_5	1	1		1	–	
x_6			1	1		–

21.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1	1	1
x_2	1	–		1		
x_3			–	1	1	
x_4	1	1	1	–		
x_5	1		1		–	1
x_6	1				1	–

22.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1	1			1
x_2	1	–		1		1
x_3	1		–	1	1	
x_4		1	1	–		
x_5			1		–	1
x_6	1		1		1	–

23.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1				1
x_2	1	–		1	1	
x_3			–	1	1	1
x_4		1	1	–		1
x_5		1	1		–	
x_6	1		1	1		–

24.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1			1	
x_2	1	–	1			1
x_3		1	–	1		1
x_4			1	–	1	
x_5	1			1	–	1
x_6		1	1		1	–

25.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1			1
x_2		–	1	1	1	
x_3	1	1	–			1
x_4		1		–		1
x_5		1			–	1
x_6	1		1	1	1	–

26.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1		1
x_2	1	–	1		1	
x_3			–	1		1
x_4		1	1	–		1
x_5		1			–	1
x_6	1		1	1		–

27.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1	1	
x_2	1	–	1			1
x_3		1	–	1	1	
x_4			1	–	1	
x_5	1		1	1	–	
x_6		1	1		1	–

28.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1		1	1	
x_2	1	–			1	1
x_3		1	–	1		1
x_4	1		1	–	1	
x_5			1	1	–	
x_6	1	1		1		–

29.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–		1			1
x_2		–	1	1	1	
x_3	1		–	1		1
x_4	1	1		–	1	
x_5	1	1		1	–	
x_6	1		1		1	–

30.

R	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	1			1	
x_2		–	1	1		1
x_3		1	–		1	1
x_4	1	1		–		1
x_5	1		1		–	1
x_6	1		1		1	–

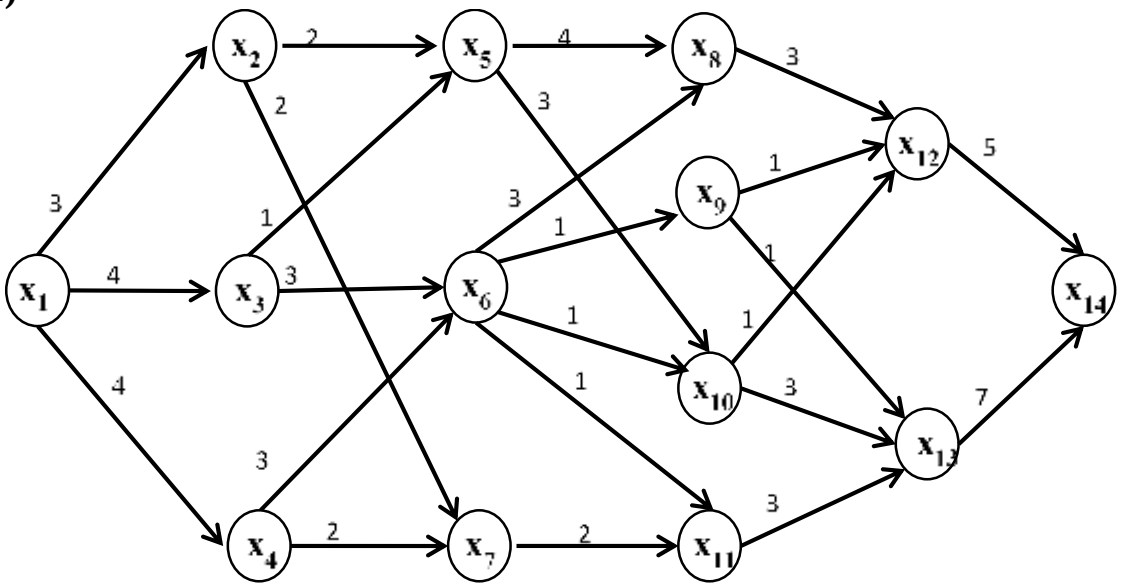
Лабораторная работа № 9
КРАТЧАЙШИЙ ПУТЬ В ГРАФЕ

Построить в графе (согласно индивидуального варианта задания):
- кратчайший путь волновым алгоритмом (для вариантов 1-10);
- кратчайший путь алгоритмом Дейкстры (для вариантов 11-20);
- длиннейший путь (для вариантов 21-30).

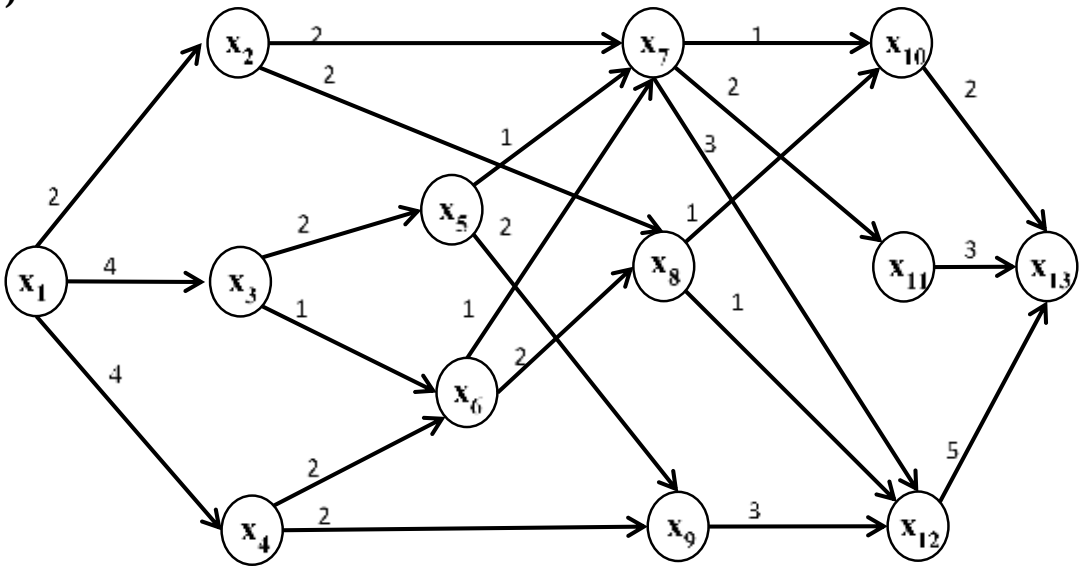
В решении должны присутствовать все промежуточные результаты.

Индивидуальные задания по вариантам

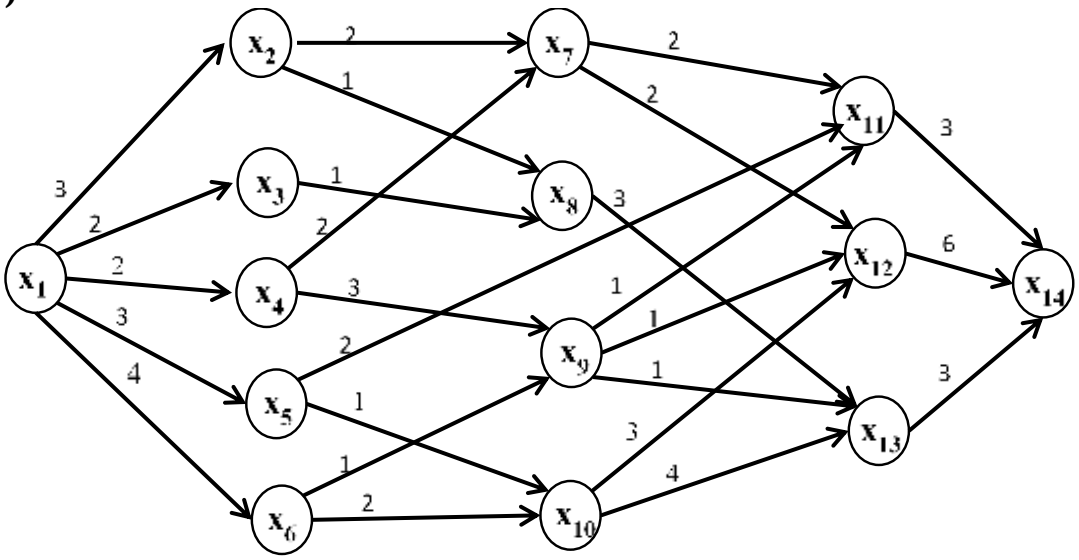
1)



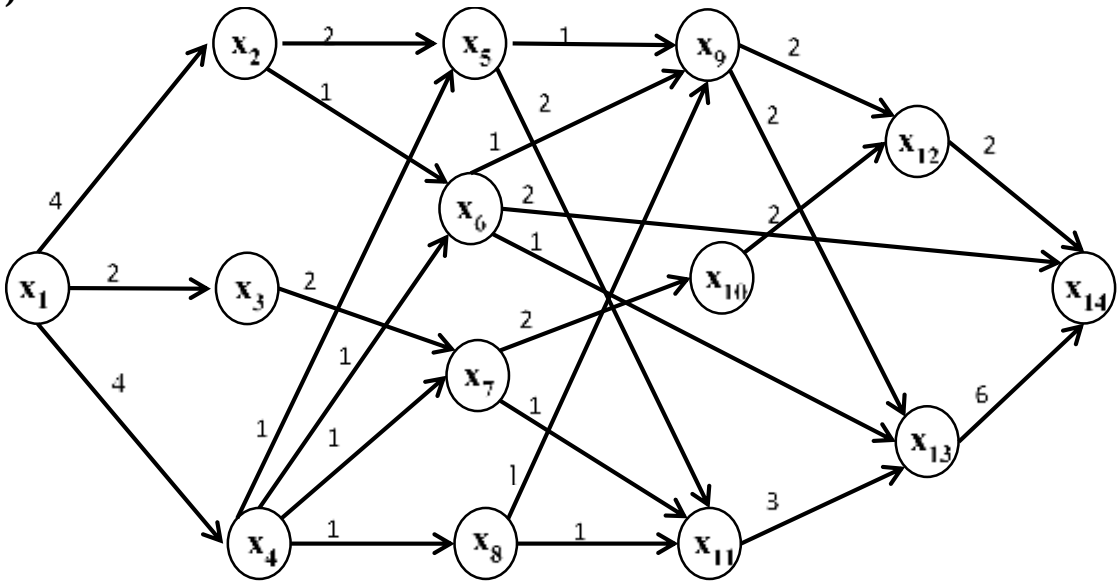
2)



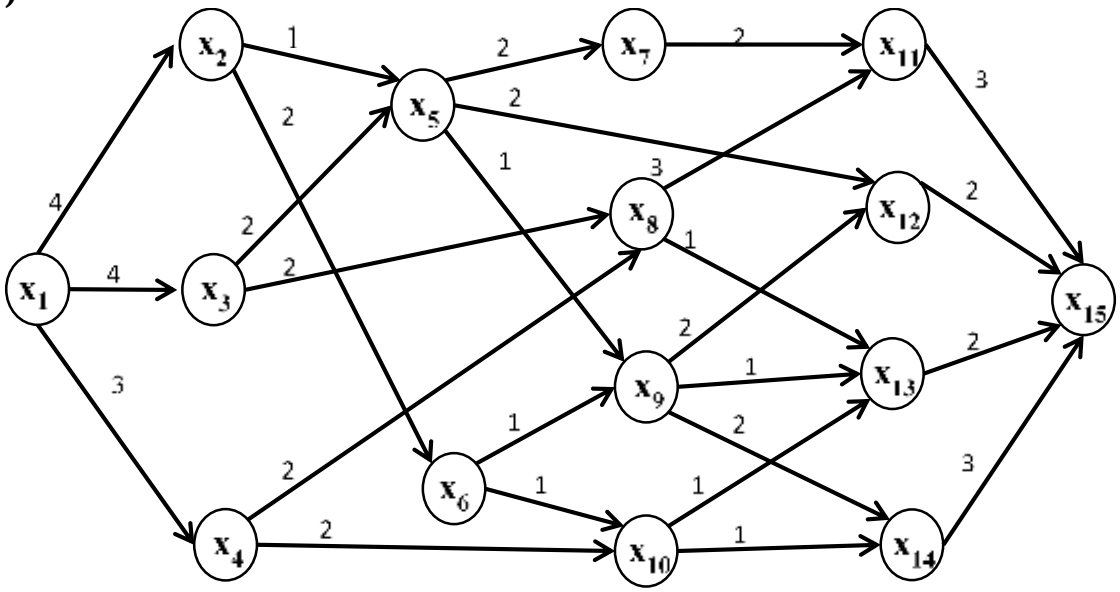
3)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

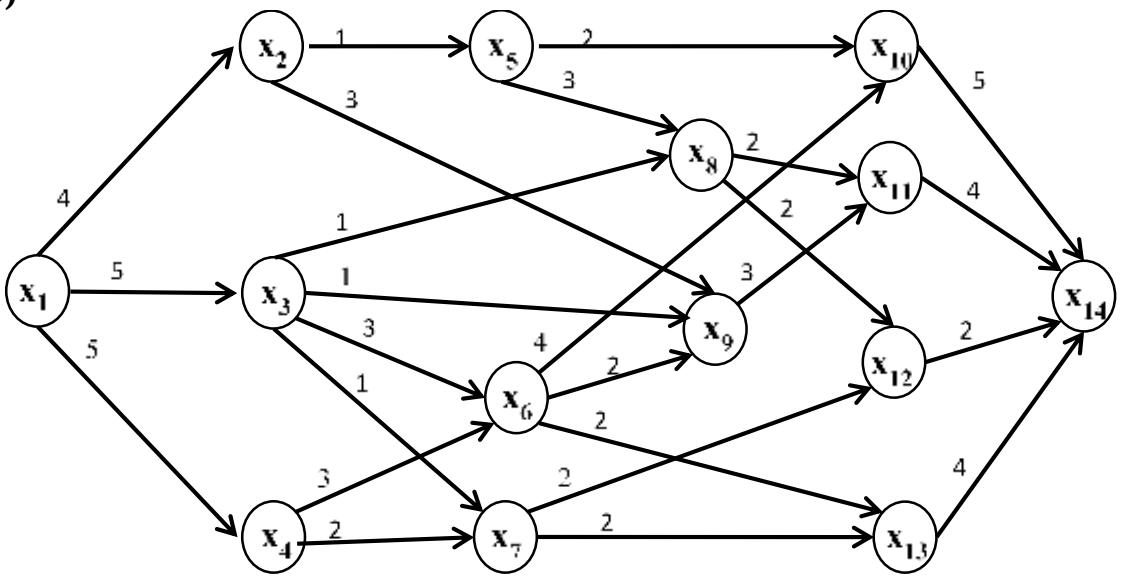
4)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

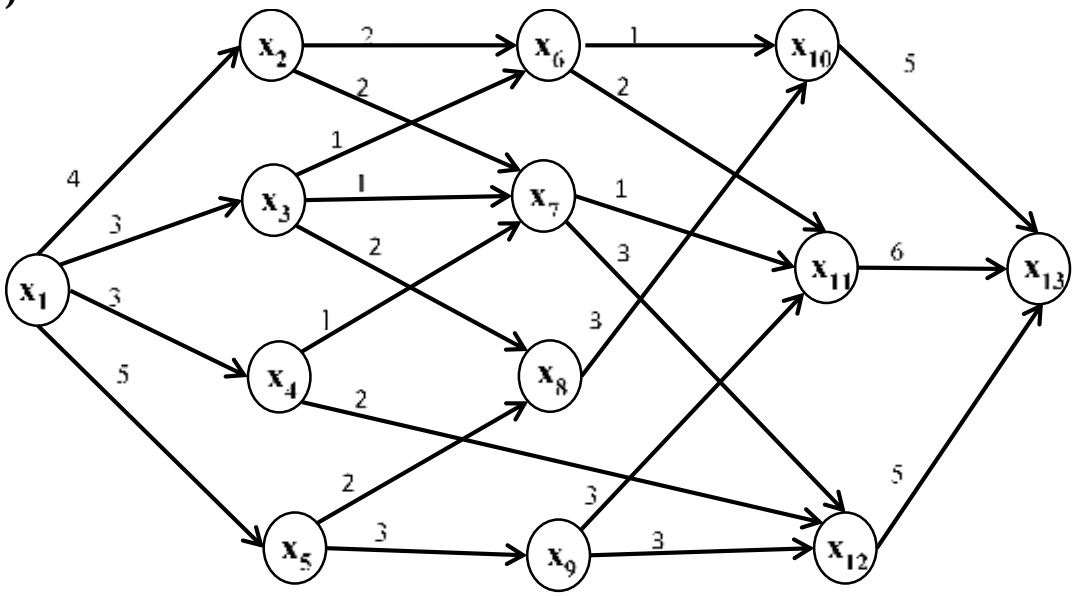
5)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
x15

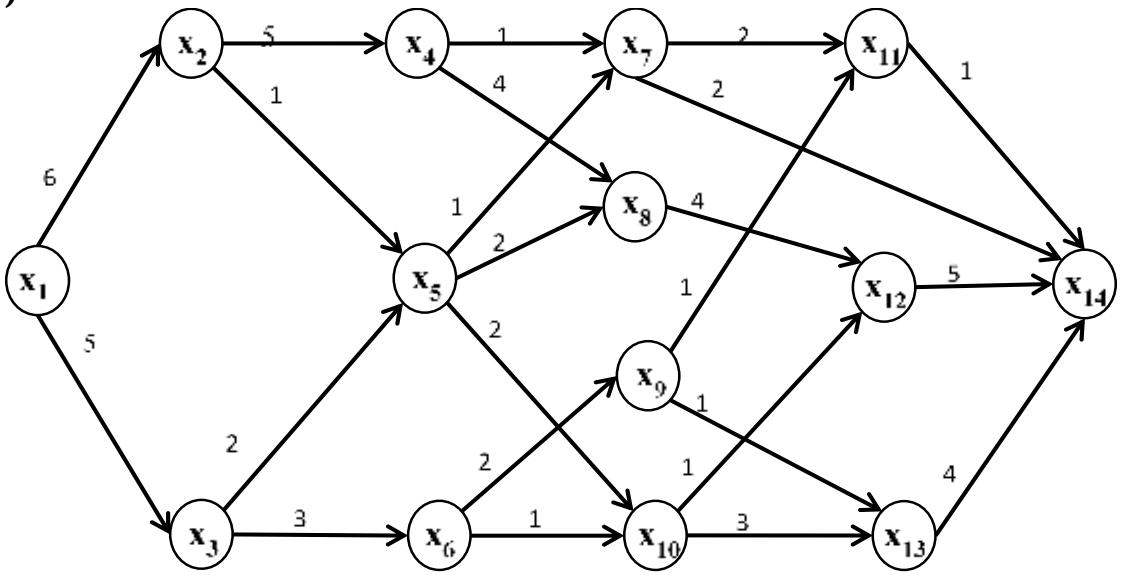
6)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

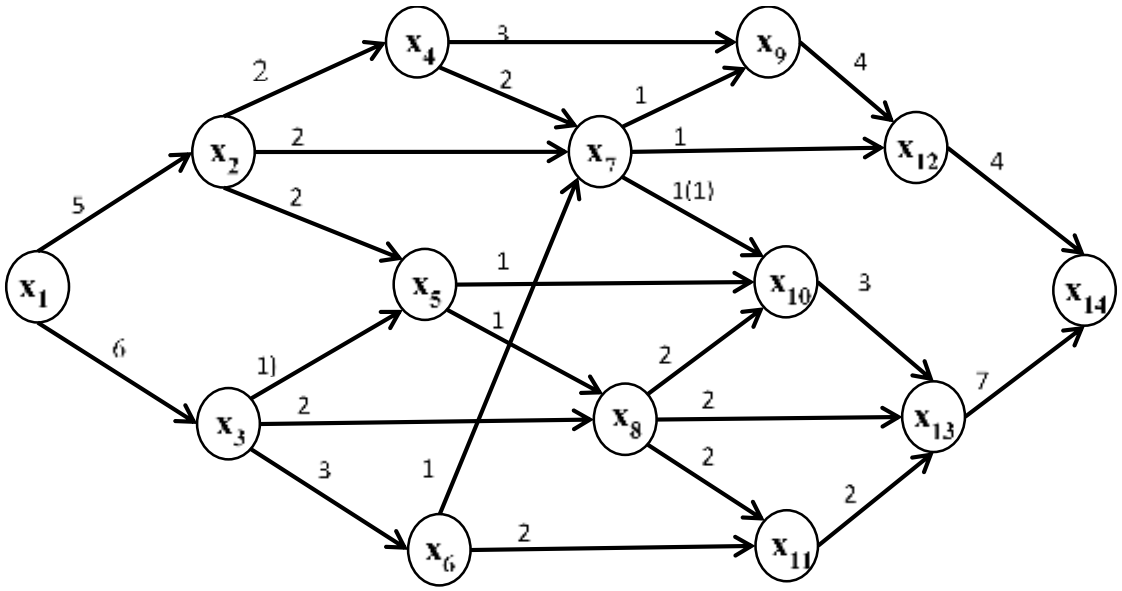
7)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

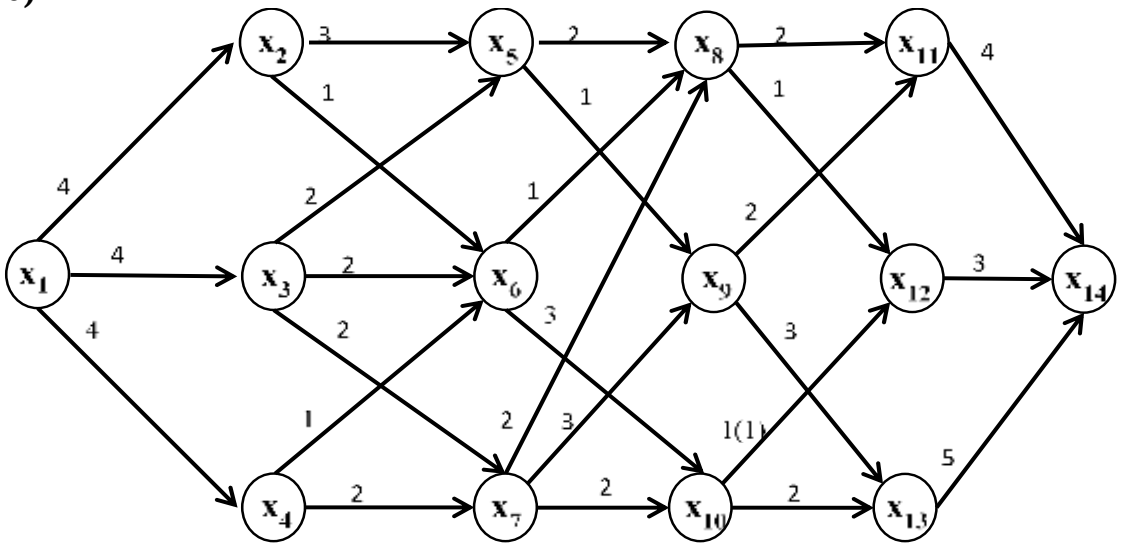
8)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

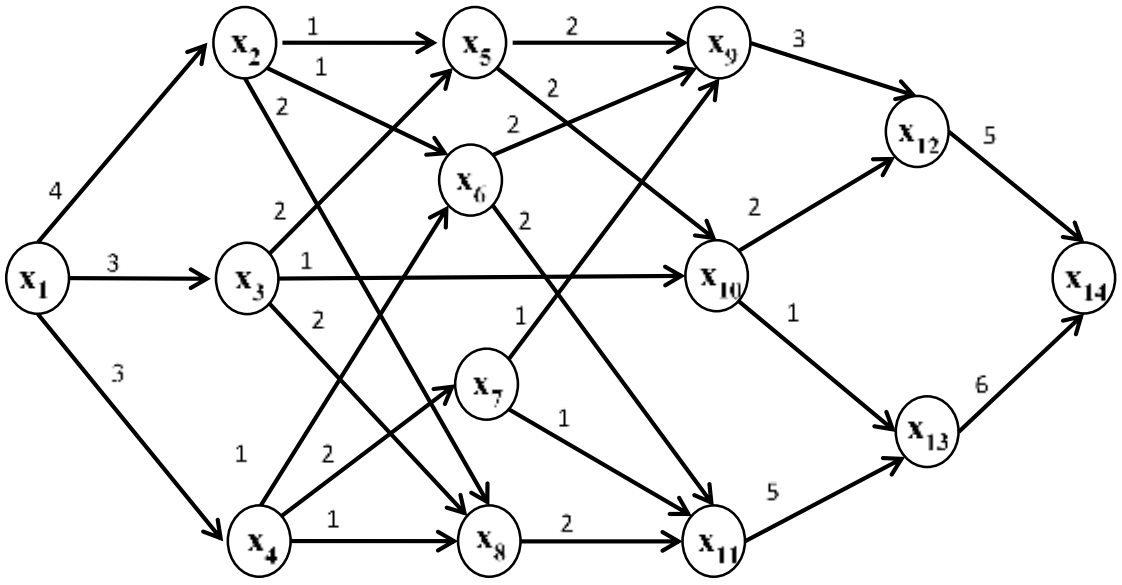
9)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
1(1)
1)

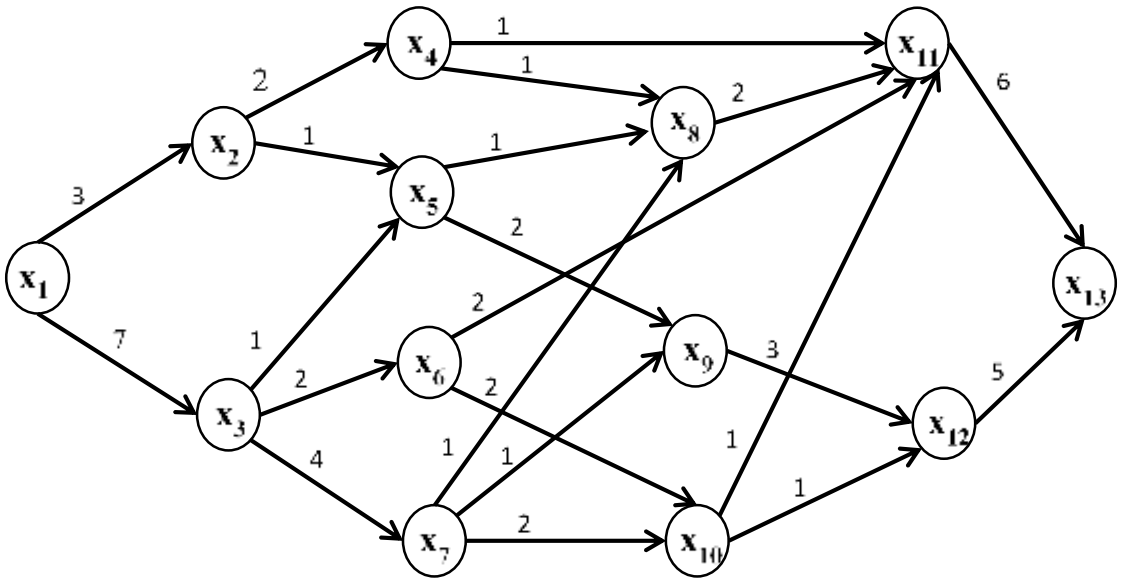
10)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
1(1)

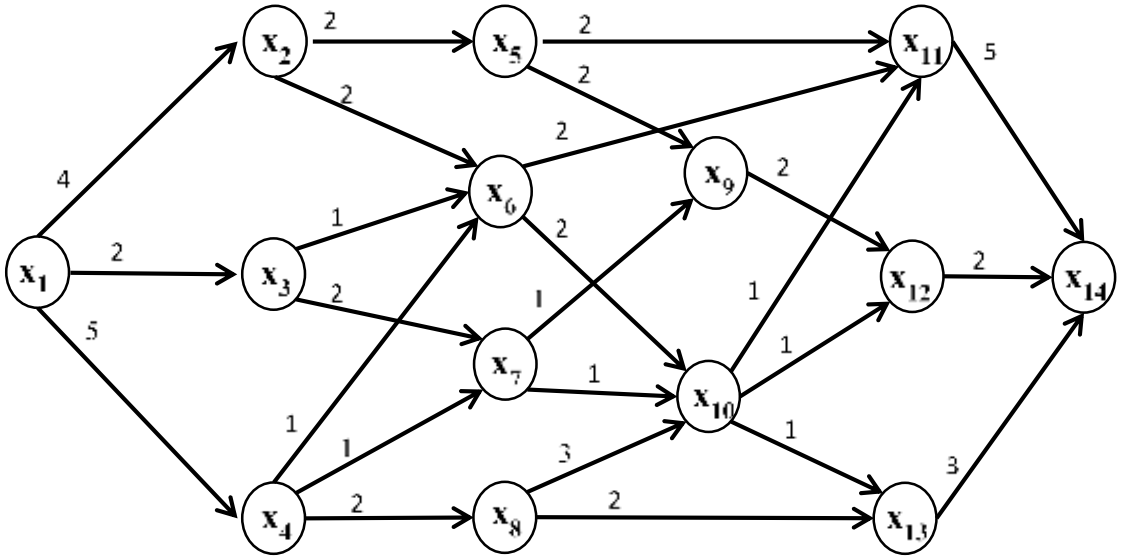
11)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

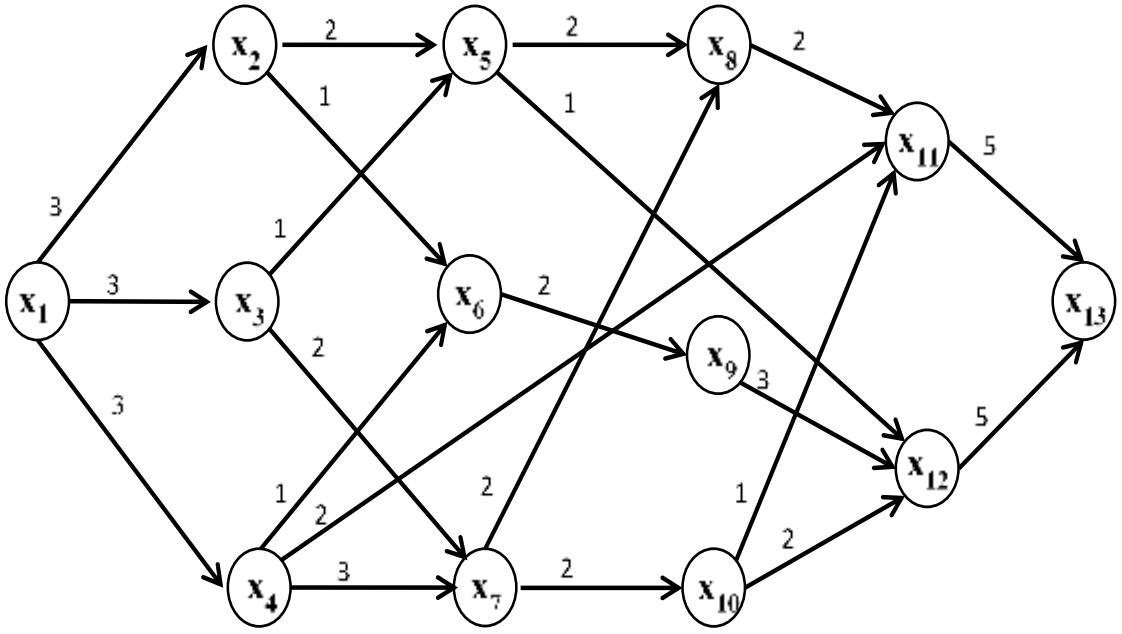
12)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

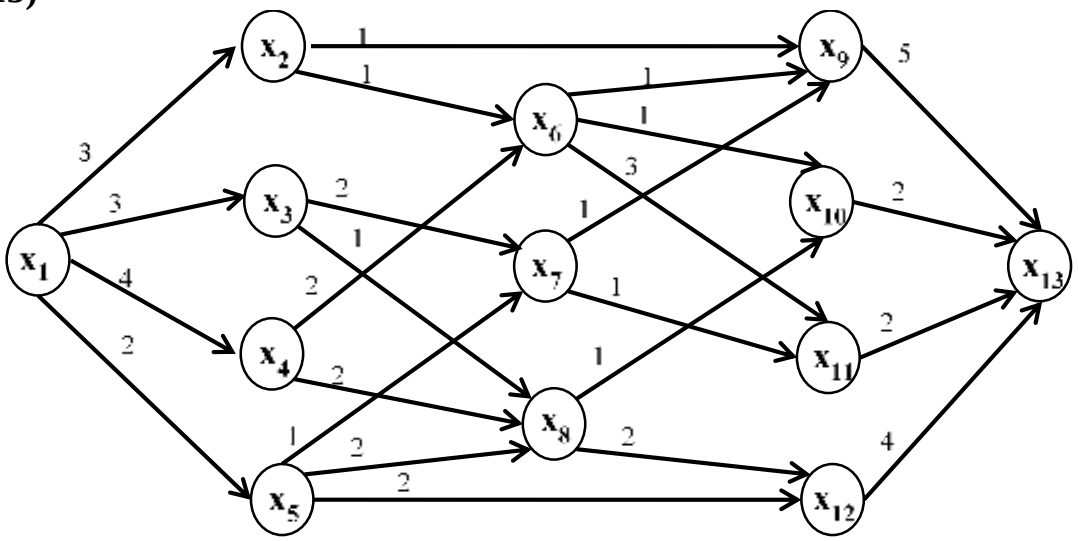
13)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

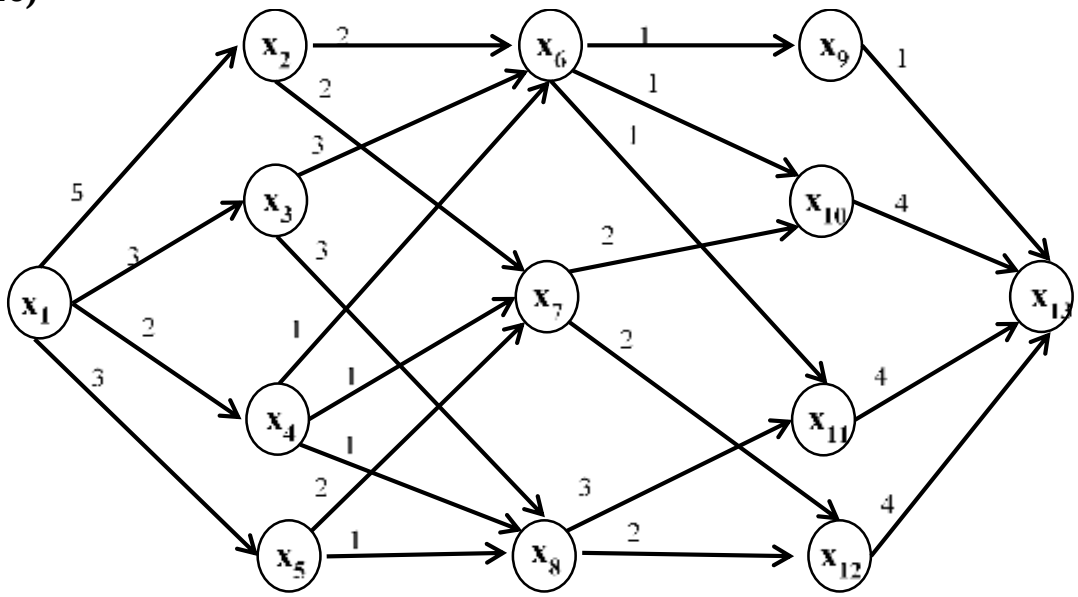
14)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

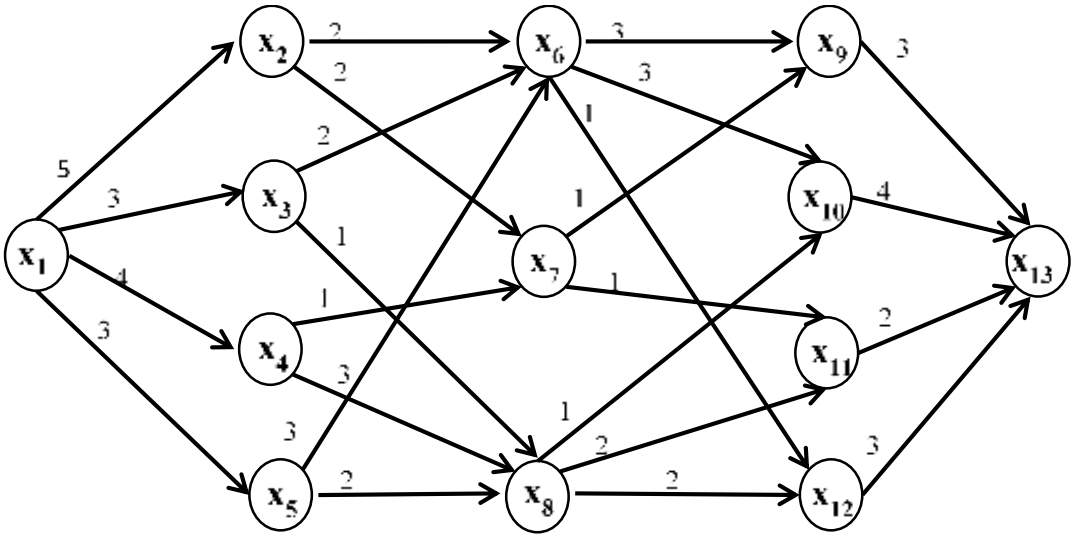
15)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

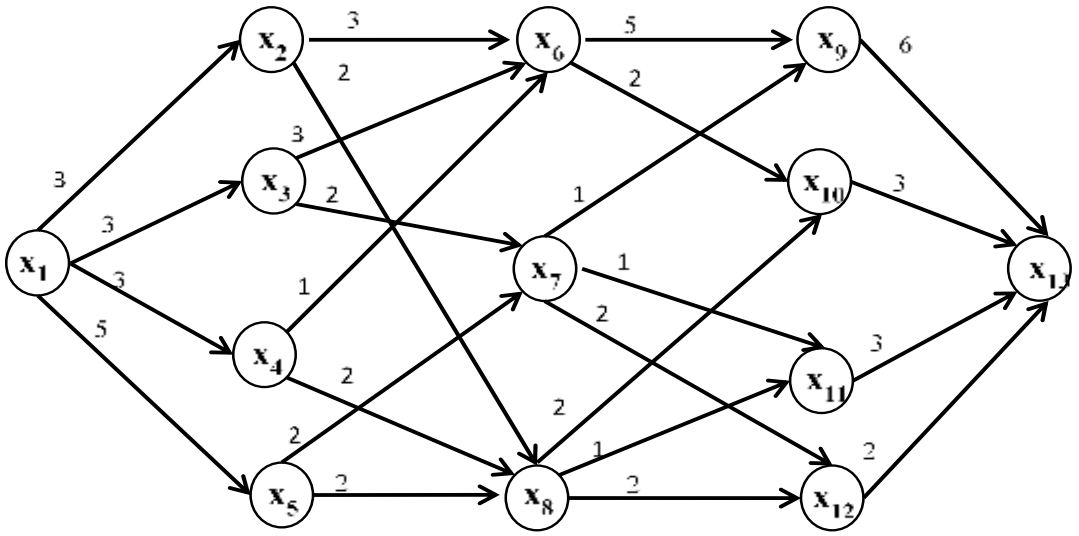
16)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

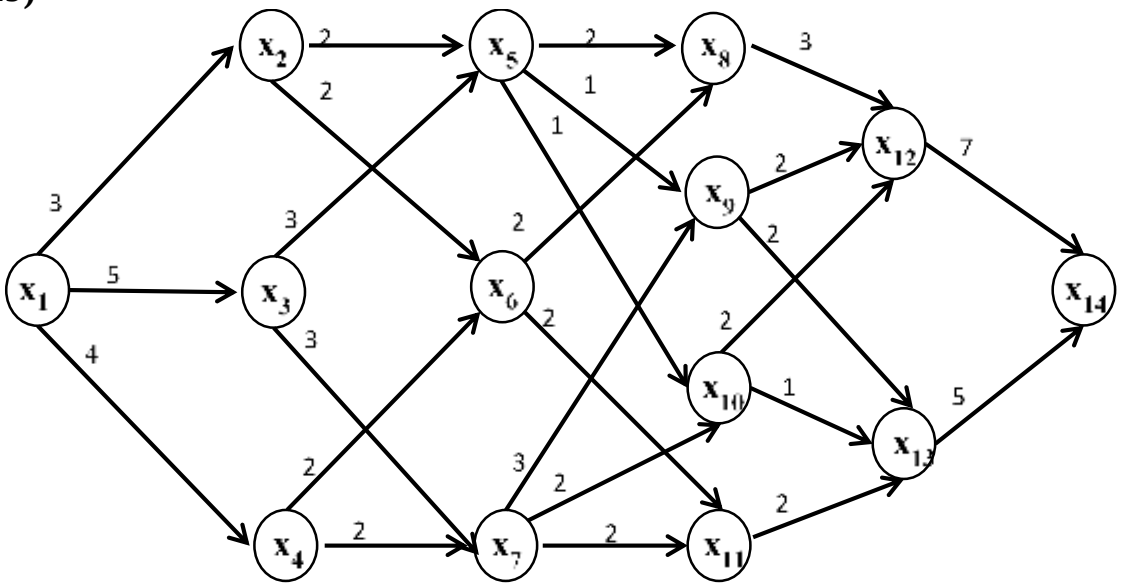
17)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

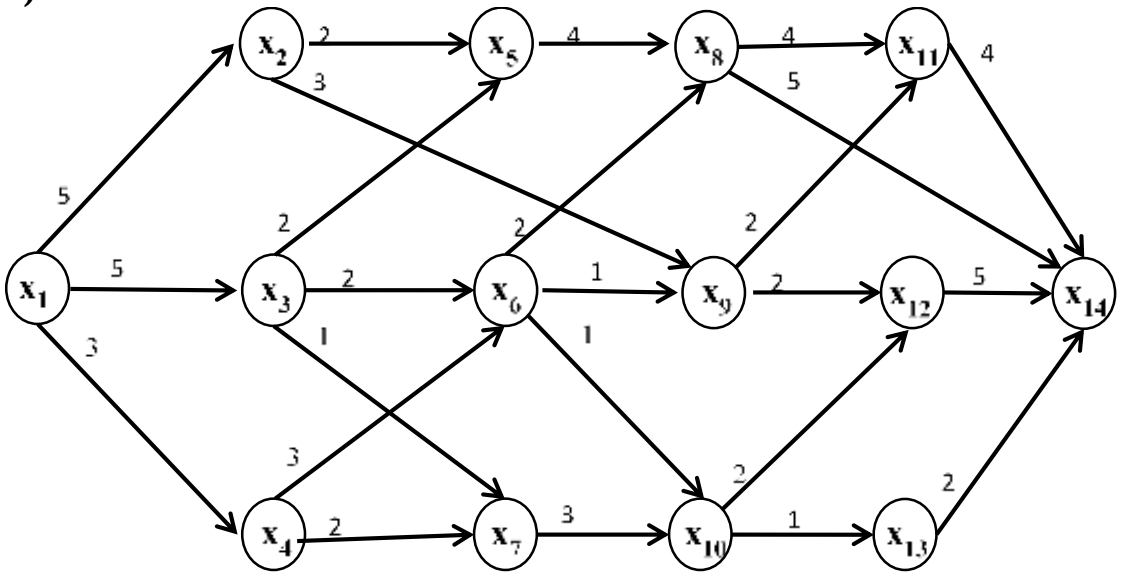
18)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

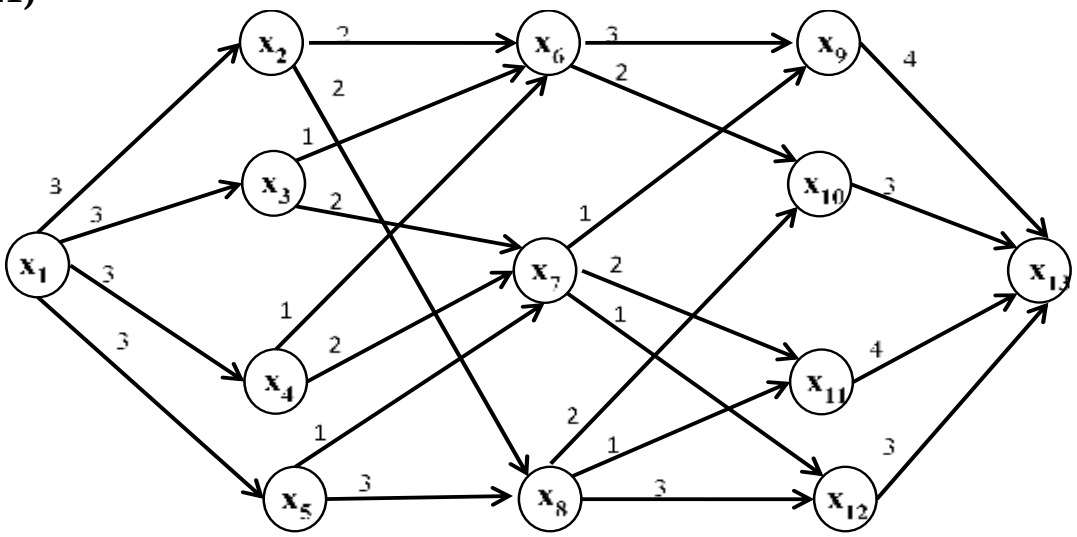
19)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

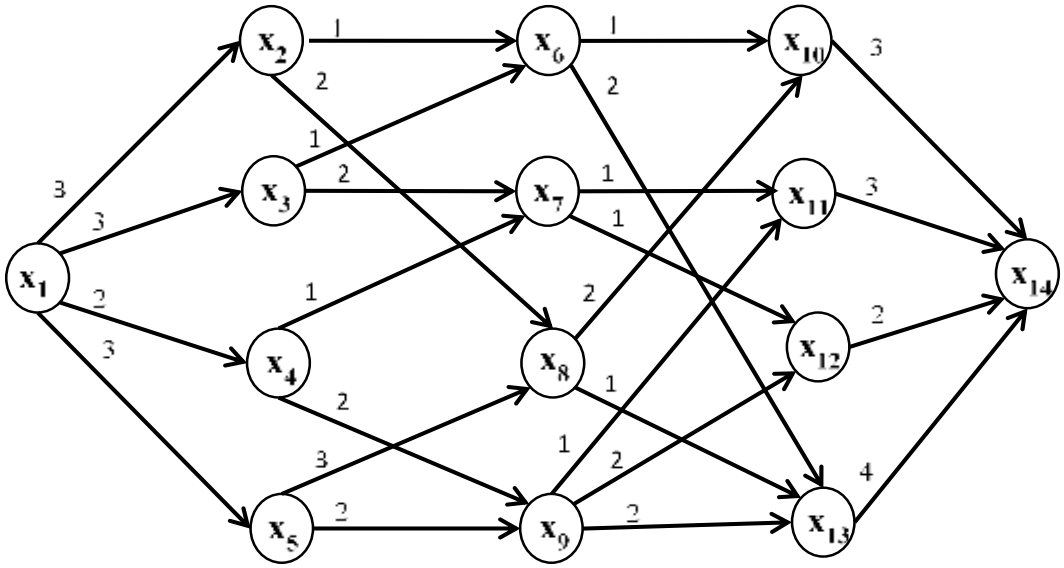
20)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

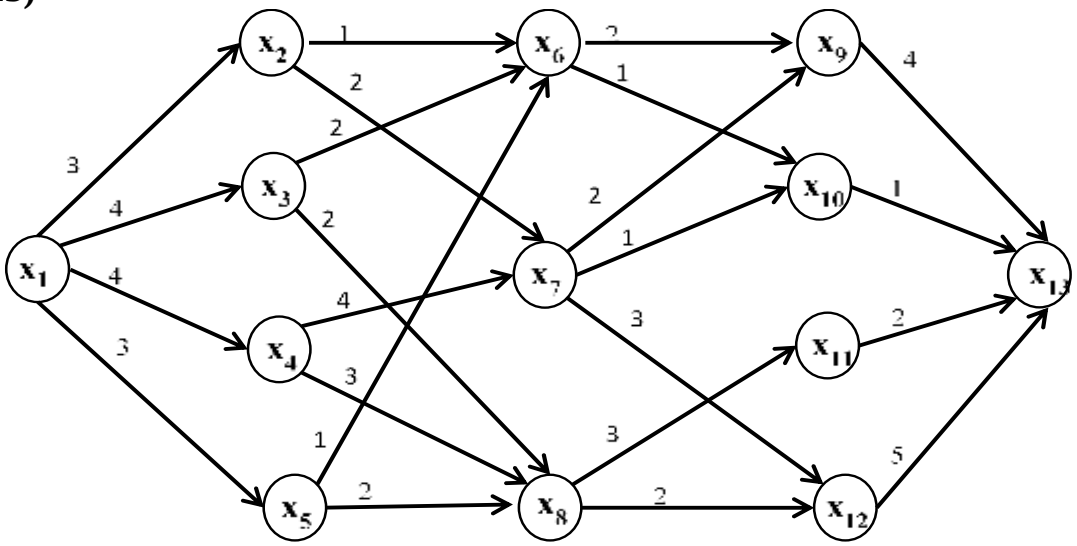
21)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

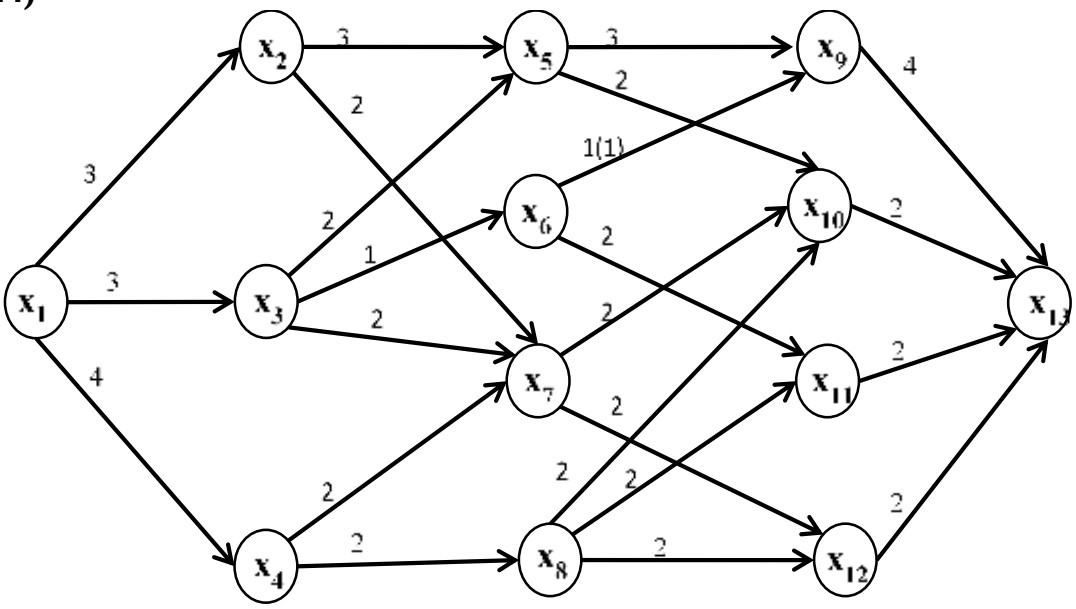
22)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

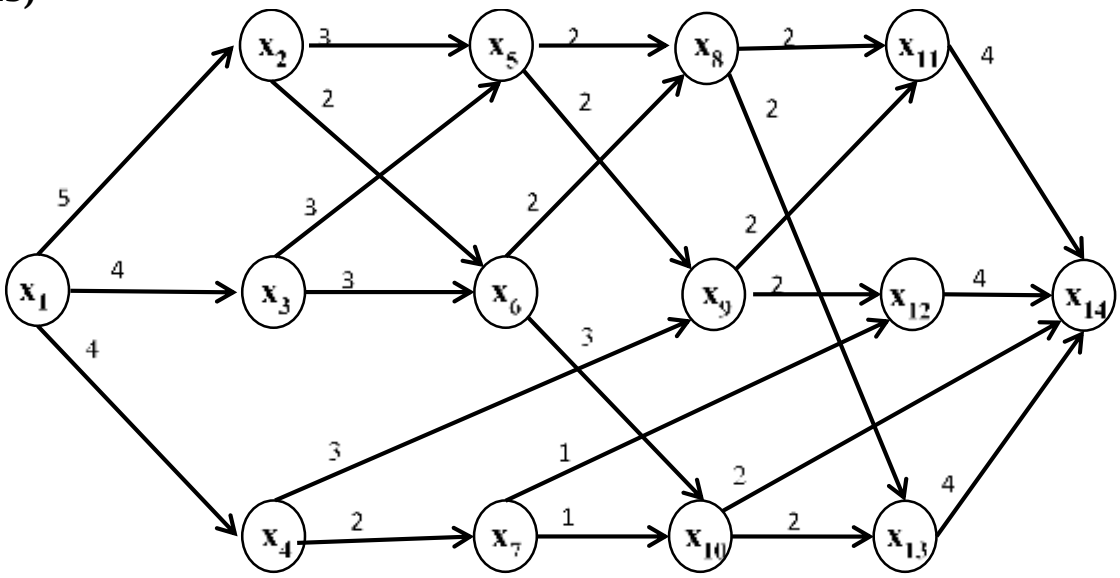
23)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

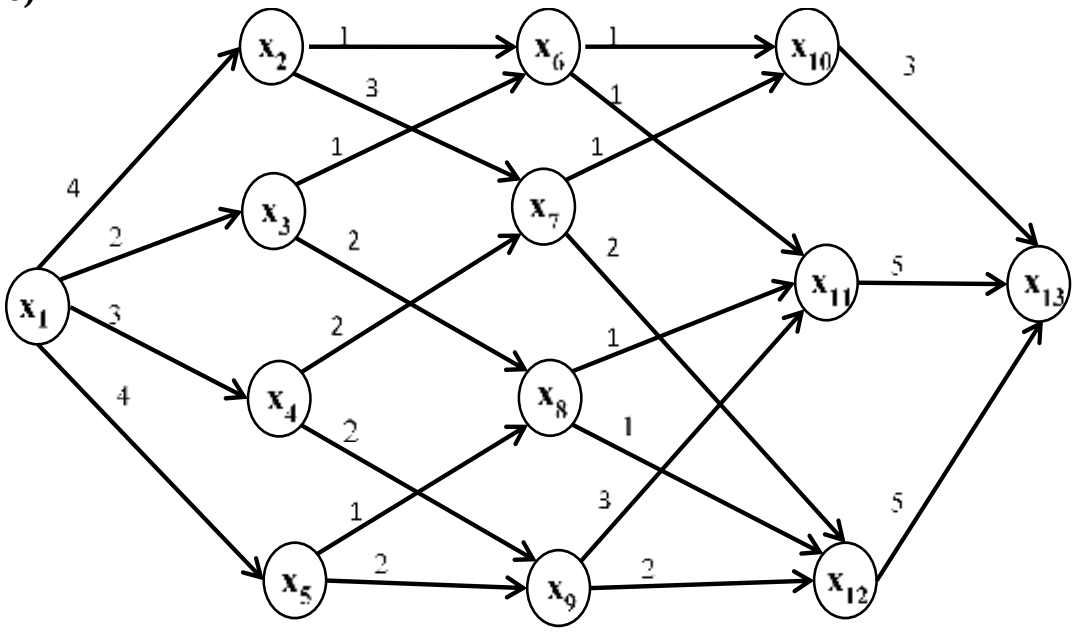
24)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
1(1)

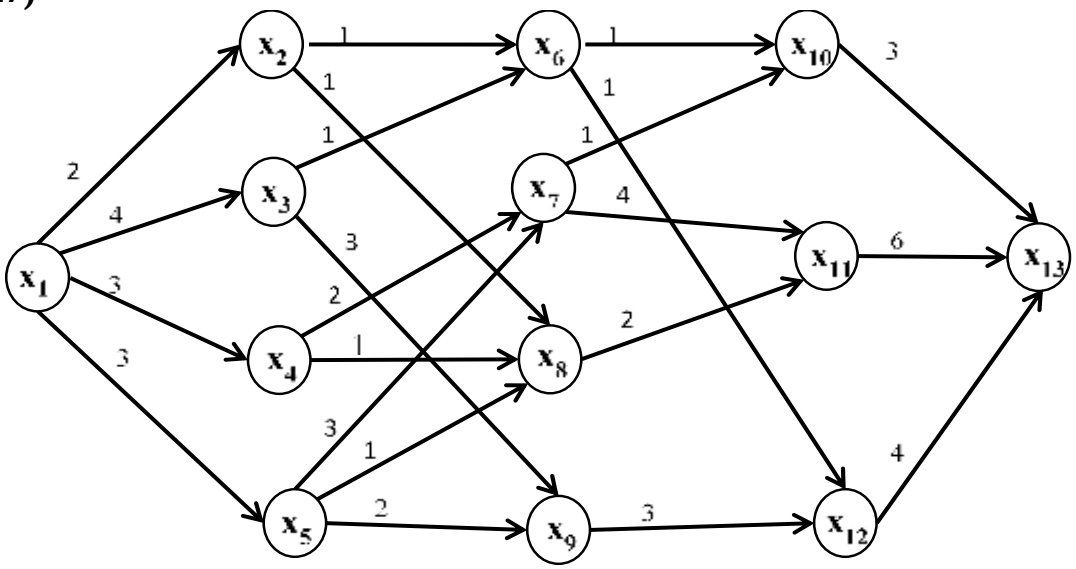
25)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14

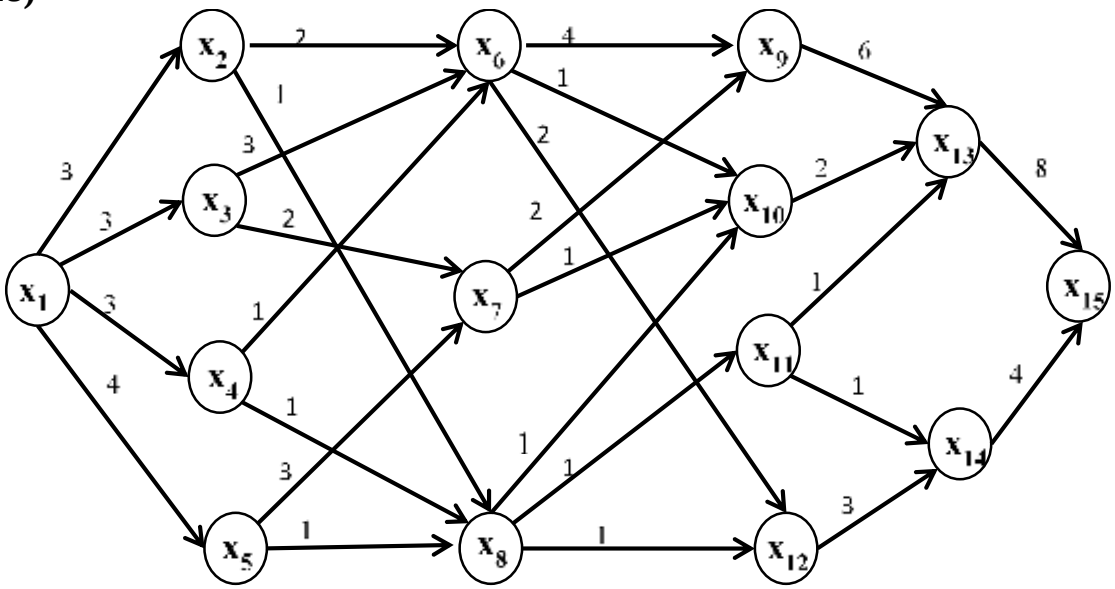
26)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

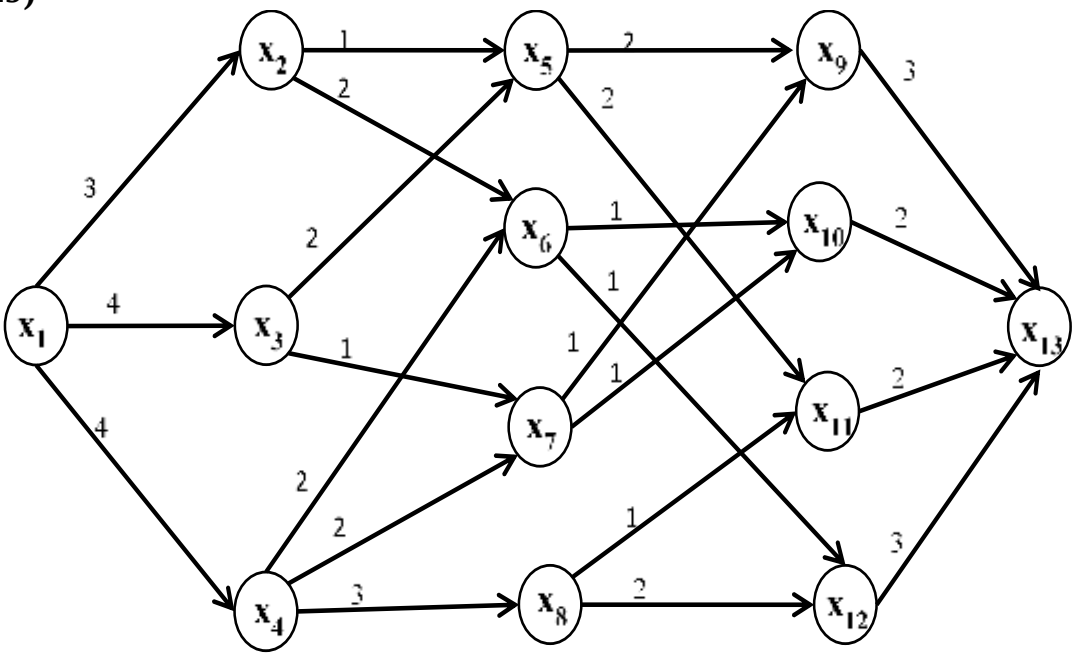
27)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

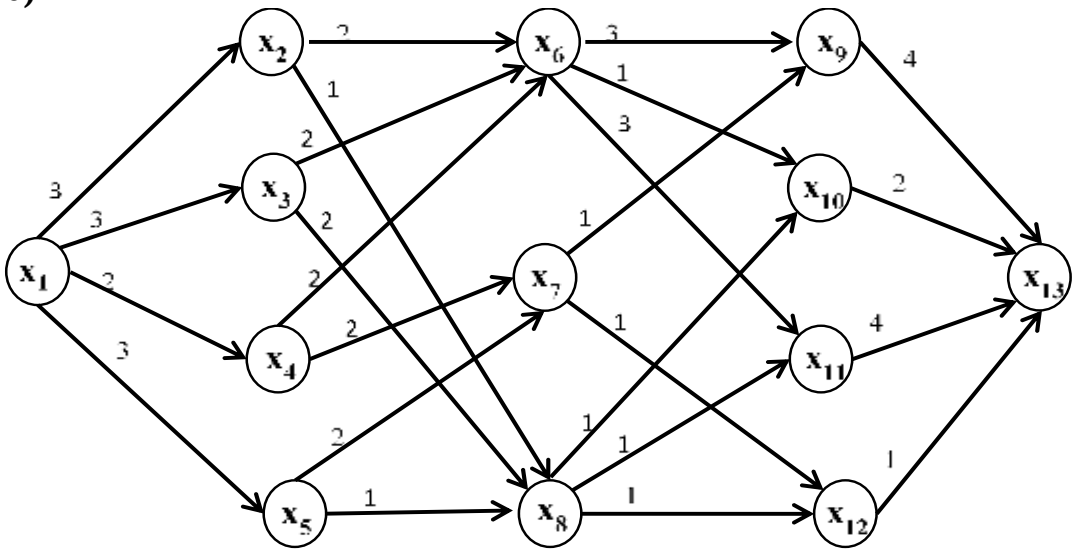
28)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
x15

29)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13

30)



Лабораторная работа №10.
ОСТОВ ГРАФА И СИСТЕМА НЕЗАВИСИМЫХ ЦИКЛОВ

Для графа (согласно индивидуального варианта задания):

1) построить минимальный остов (с обязательной нумерацией каждого шага);

2) определить цикломатическое число графа;

3) записать систему независимых циклов (запись каждого цикла начинать с ребра, не принадлежащего остову).

Индивидуальные задания по вариантам

1.

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

2.

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

3.

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

4.

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

5.

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

6.

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

7.

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

8.

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

9.

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

10.

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

11.

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

12.

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

13.

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

14.

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

15.

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

16.

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

17.

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

18.

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

19.

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

20.

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

21.

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

22.

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

23.

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

24.

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

25.

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

26.

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

27.

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

28.

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

29.

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

30.

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

Лабораторная работа № 11.
КРАТЧАЙШАЯ РАСКРАСКА ВЕРШИН В ГРАФЕ

Для графа (согласно индивидуального варианта задания):

1) определить хроматическое число графа;
2) произвести кратчайшую раскраску вершин графа.

В решении должны присутствовать и быть описаны все промежуточные результаты.

Индивидуальные задания по вариантам

1.

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

2.

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

3.

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

4.

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

5.

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

6.

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

7.

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

8.

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

9.

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

10.

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

11.

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

12.

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

13.

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

14.

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

15.

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

16.

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

17.

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

18.

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

19.

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

20.

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

21.

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

22.

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

23.

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

24.

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

25.

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

26.

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

27.

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

28.

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

29.

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

30.

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

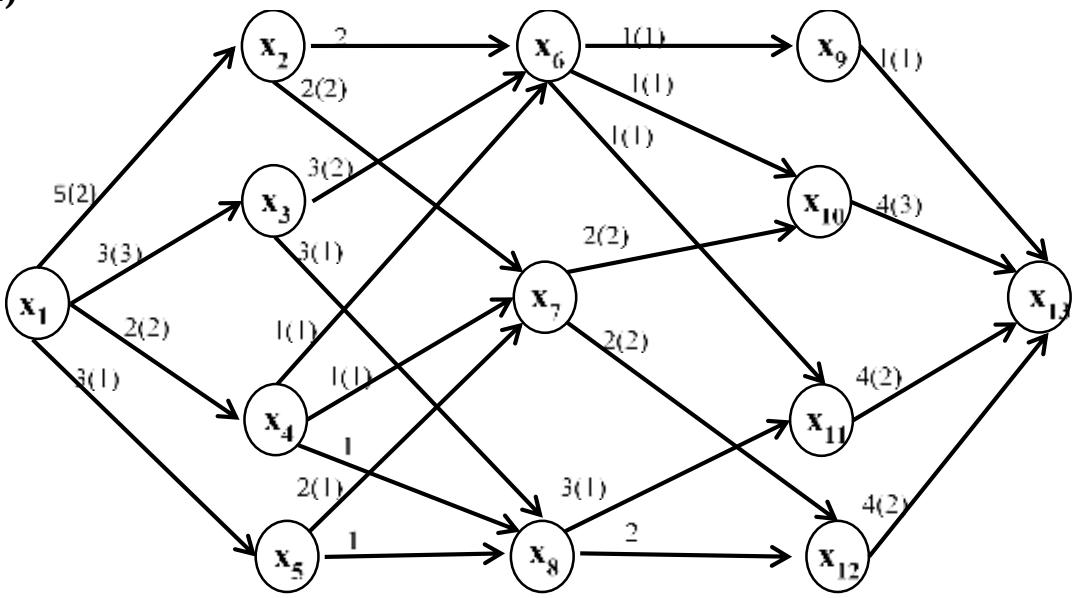
Лабораторная работа №12
МАКСИМАЛЬНЫЙ ПОТОК В СЕТИ

Для графа (согласно индивидуального варианта задания) построить максимальный поток в сети.

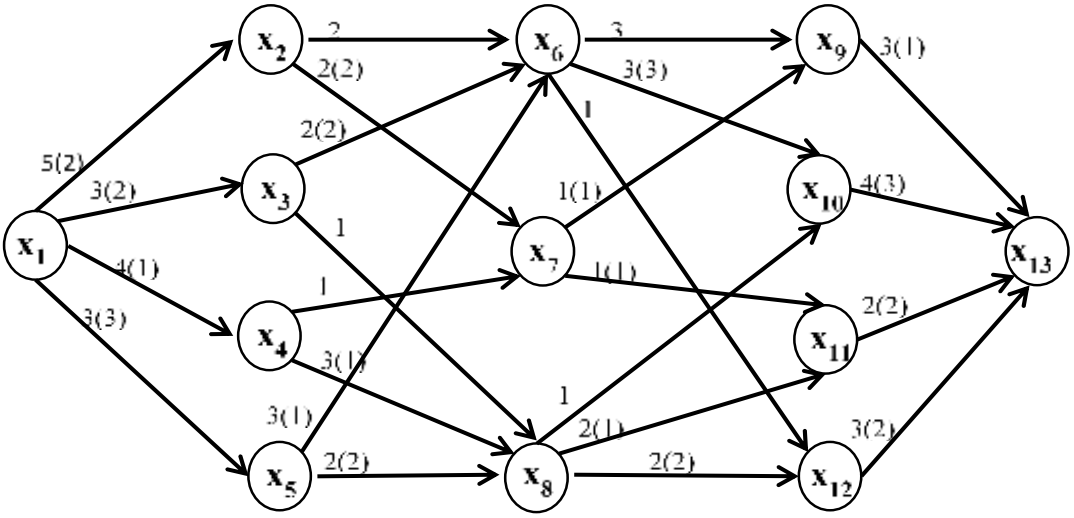
В решении должны присутствовать все промежуточные результаты.

Индивидуальные задания по вариантам

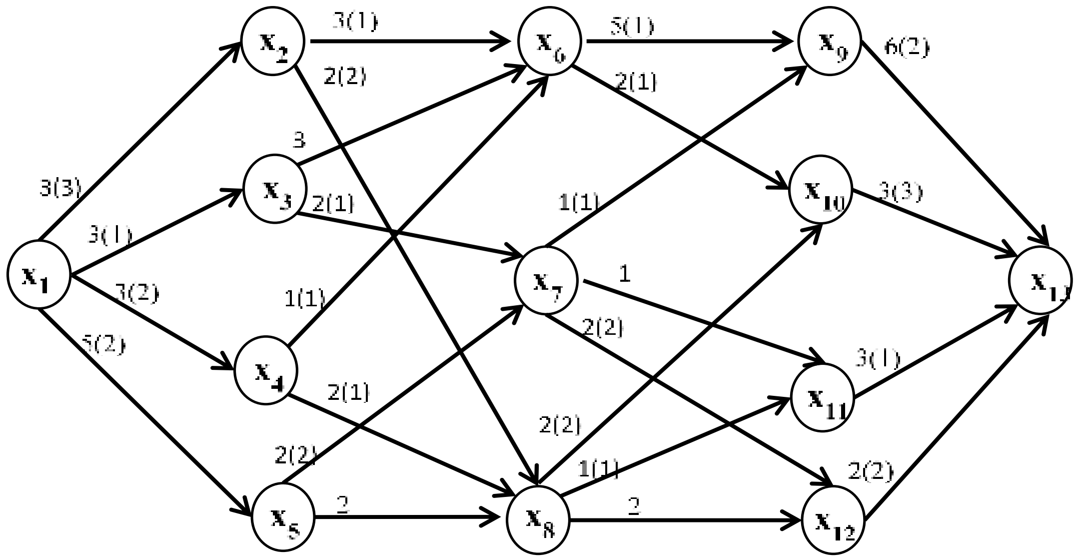
1)



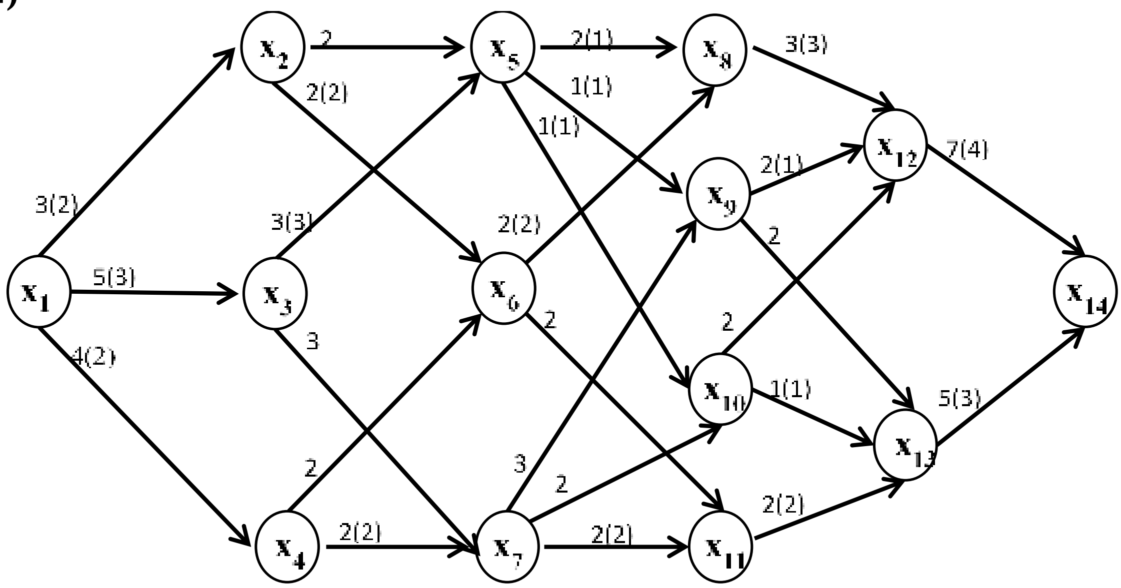
2)



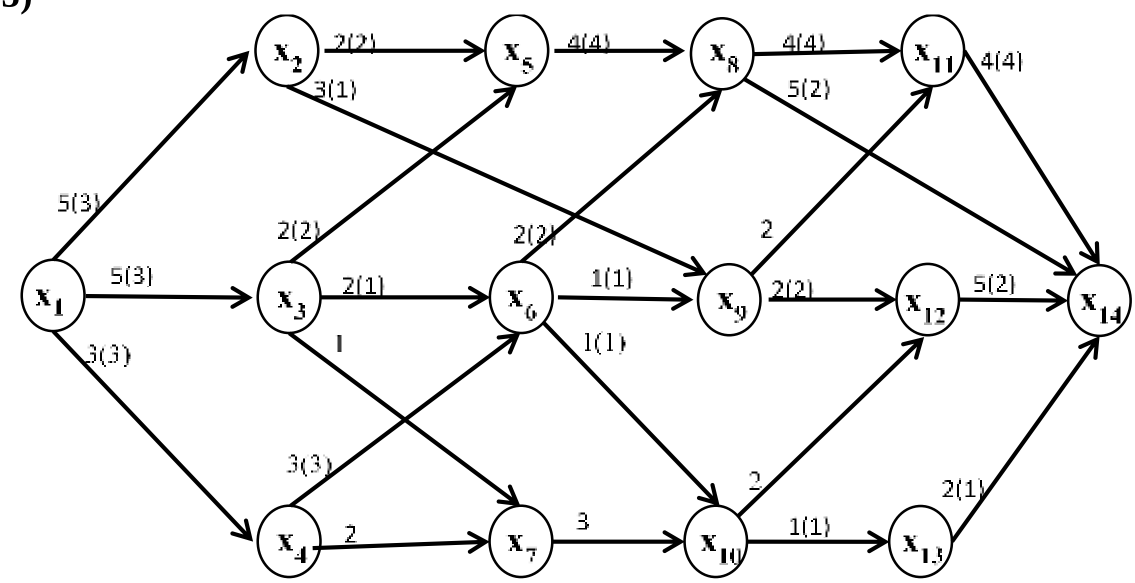
3)


x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
3(3)
3(1)
3(2)
5(2)
3(1)
2(2)
3
2(1)
1(1)
2(1)
2(2)
2
5(1)
2(1)
1(1)
1
2(2)
2(2)
1(1)
2
6(2)
3(3)
3(1)
2(2)

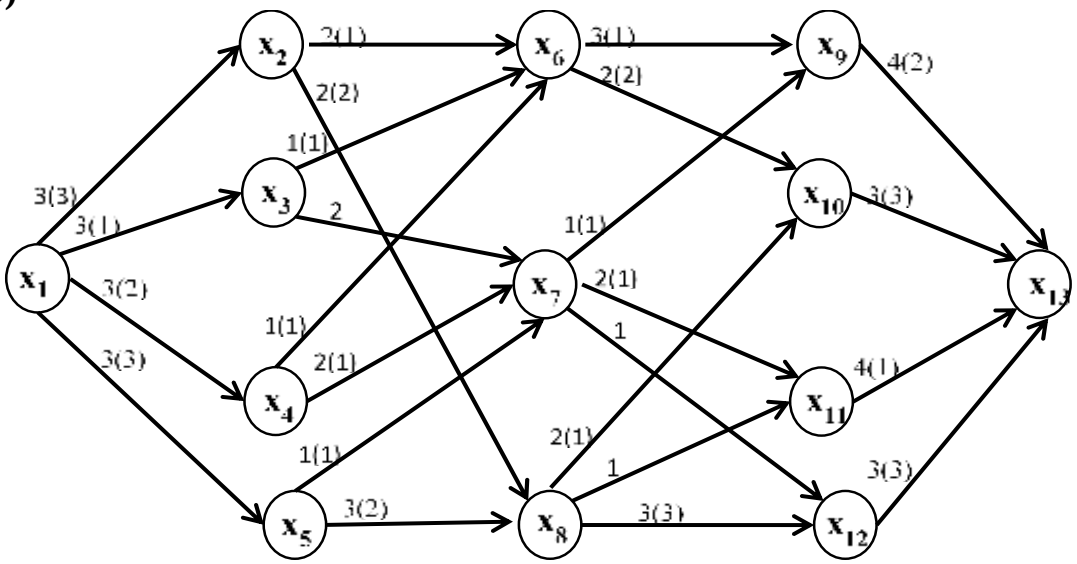
4)


x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
3(2)
5(3)
4(2)
2
2(2)
3(3)
3
2
2(2)
2(1)
1(1)
1(1)
2(2)
2
3
2
2(2)
3(3)
2(1)
2
2
1(1)
2(2)
7(4)
5(3)

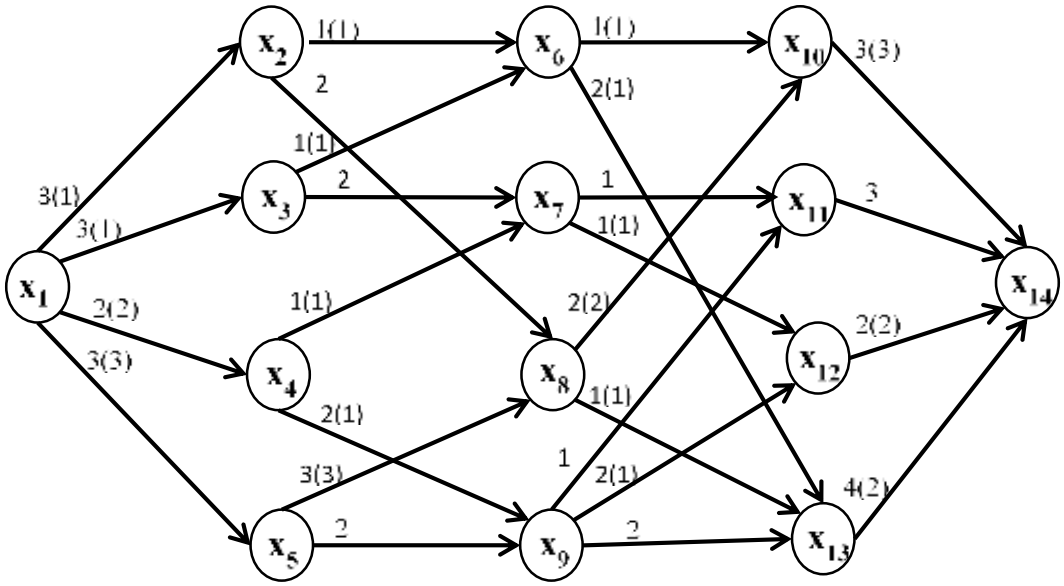
5)


x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
5(3)
5(3)
3(3)
2(2)
3(1)
2(2)
2(1)
1
3(3)
2
4(4)
2(2)
1(1)
1(1)
3
4(4)
5(2)
2
2(2)
2
1(1)
4(4)
5(2)
2(1)

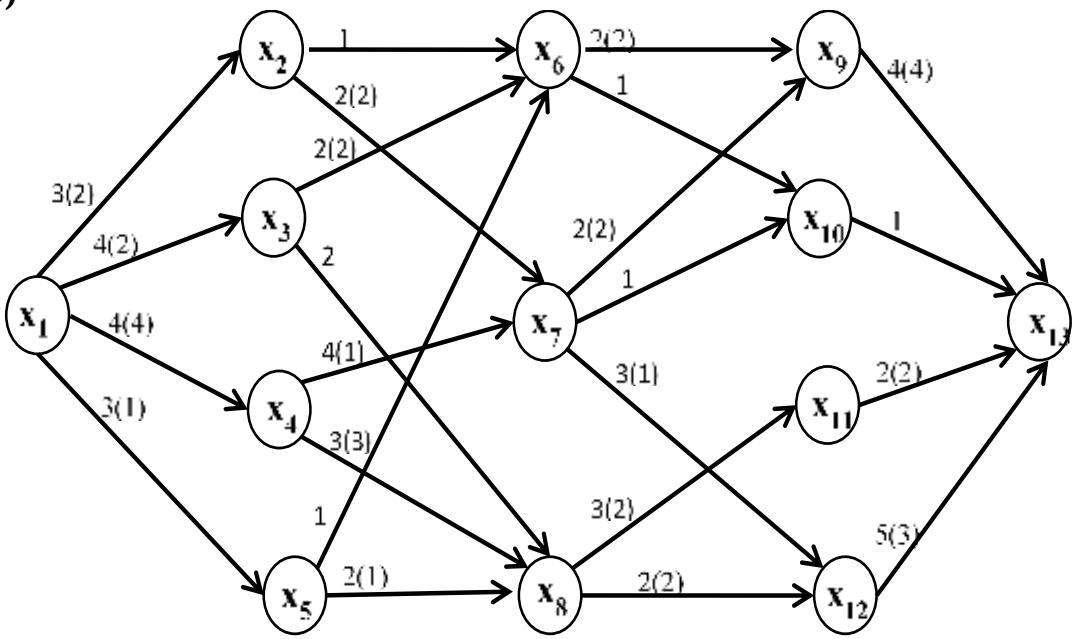
6)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(3)
3(1)
3(2)
3(3)
2(1)
2(2)
1(1)
2
1(1)
2(1)
1(1)
3(2)
2(1)
1
1
3(3)
2(1)
4(2)
2(2)
3(3)
4(1)
3(3)
3(3)

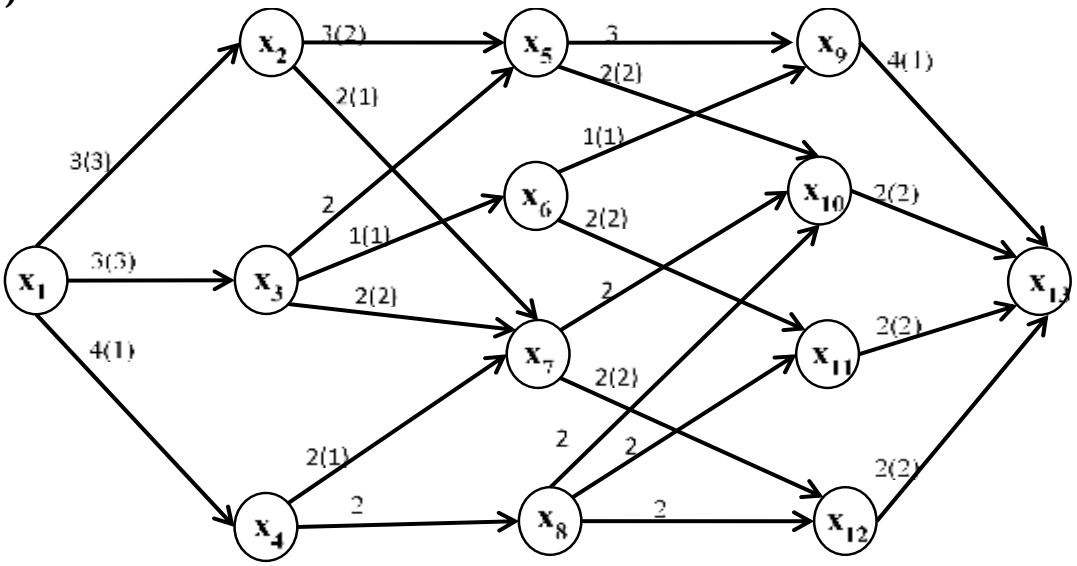
7)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
3(1)
3(1)
2(2)
3(3)
1(1)
2
1(1)
2(1)
3(3)
2
1(1)
2(2)
1
1(1)
1
2(1)
2
3(3)
2(1)
2(2)
1(1)
2(2)
4(2)
2
2

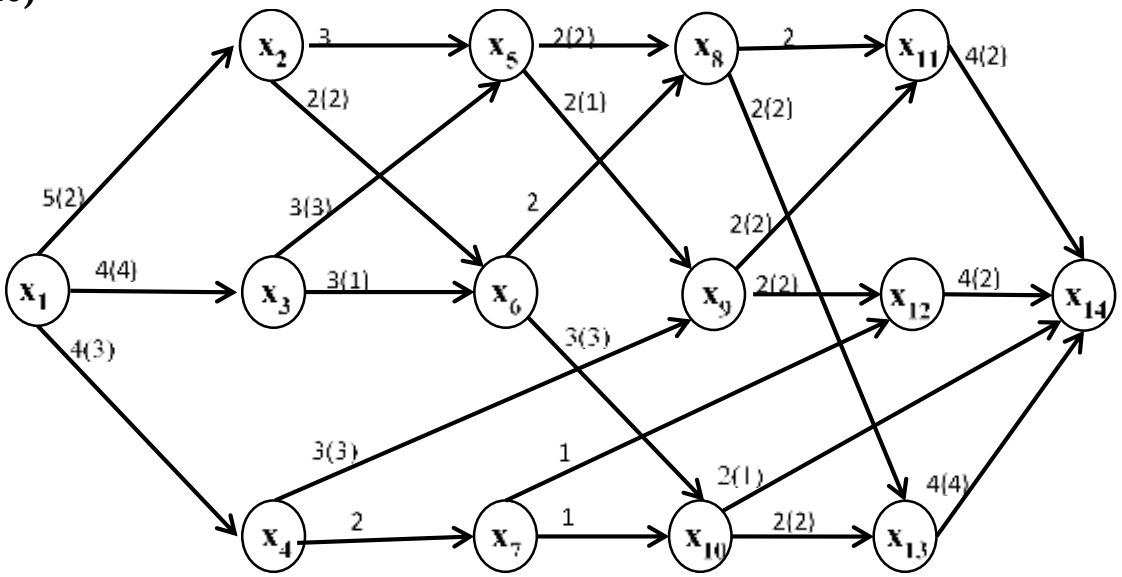
8)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(2)
4(2)
4(4)
3(1)
1
2(2)
2(2)
2
4(1)
3(3)
1
2(2)
1
3(1)
3(2)
2(1)
2(2)
4(4)
1
2(2)
3(1)
5(3)

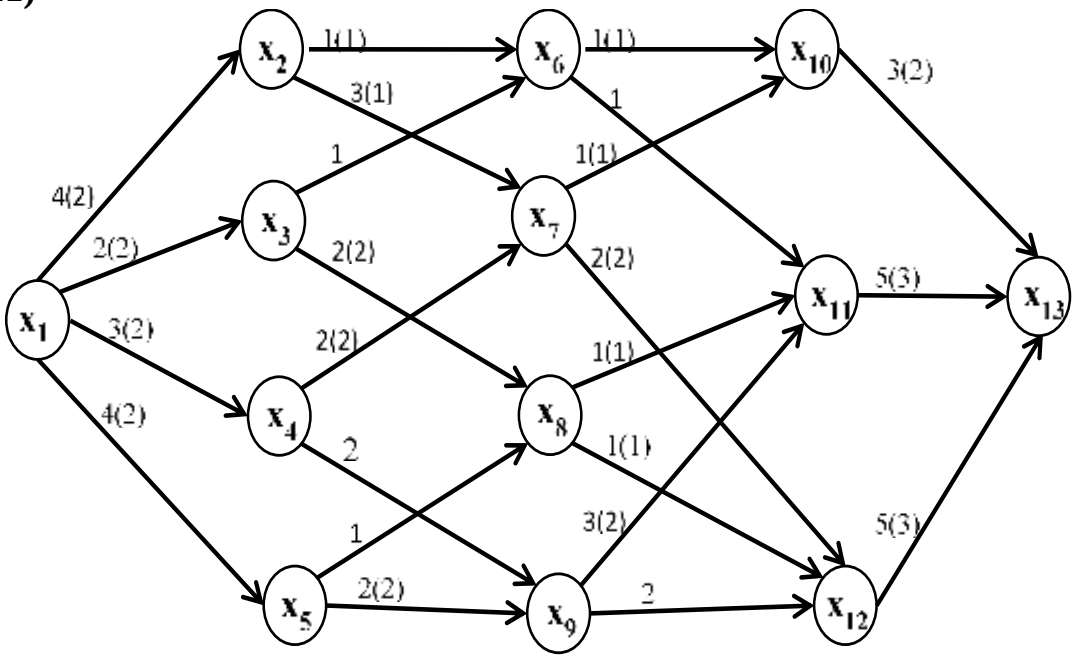
9)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(3)
4(1)
3(2)
2(1)
2
1(1)
2(2)
3
4(1)
2(2)
2(2)
2
2(2)
2(1)
2
2
2
2
2(2)
2
2

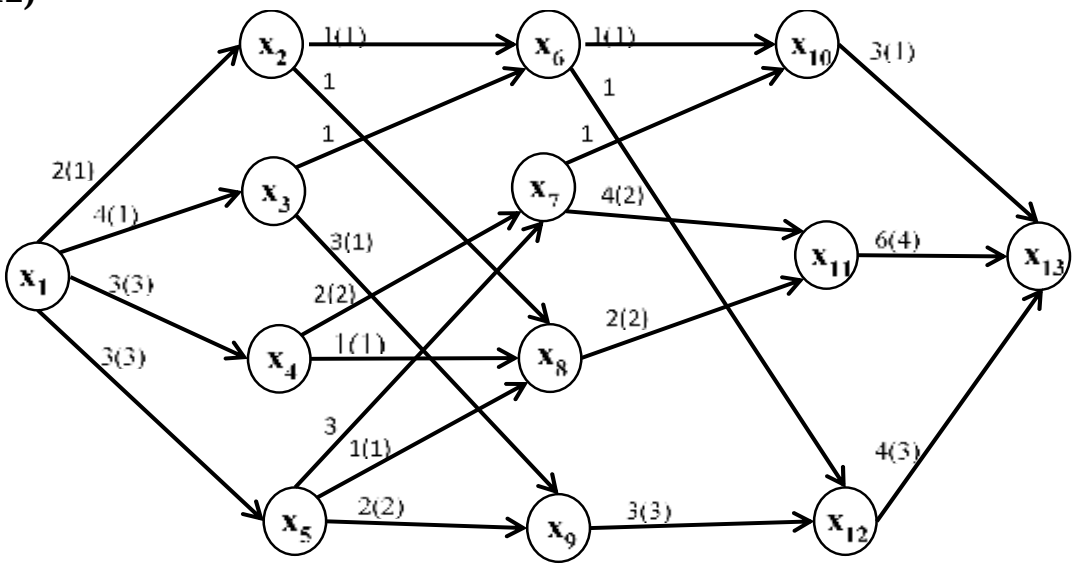
10)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x14
5(2)
4(4)
4(3)
3
2(2)
3(3)
3(1)
2(2)
2
2(1)
2
2(2)
2(2)
4(2)
2(2)
4(2)
3(3)
3(3)
1
2
1
2(1)
2(2)
4(4)

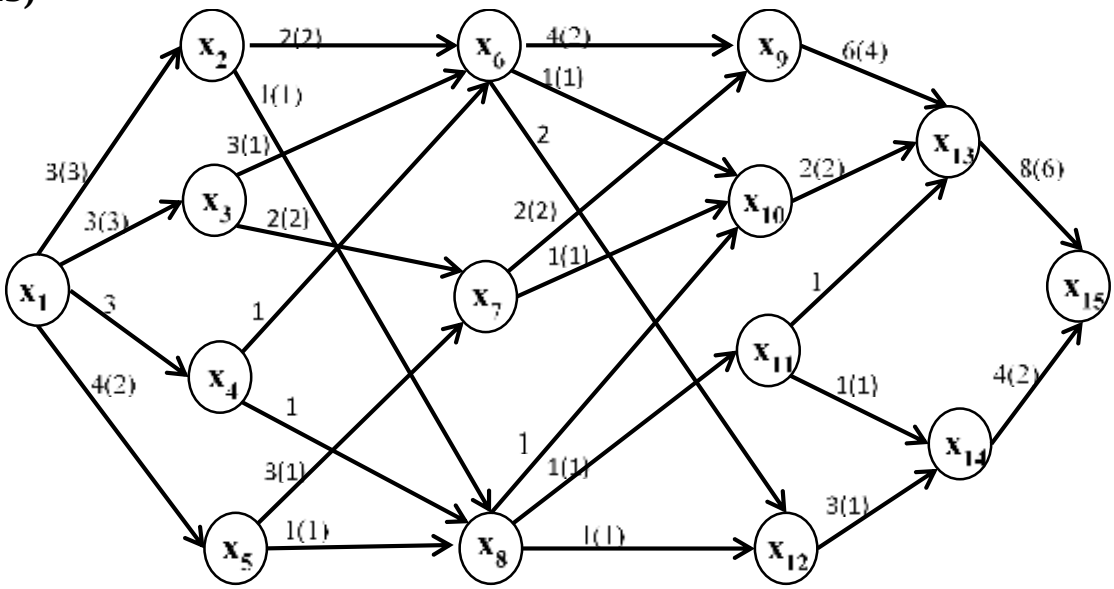
11)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
4(2)
2(2)
3(2)
4(2)
1(1)
3(1)
1
2(2)
2(2)
1
1(1)
1
2(2)
1(1)
2
1
2(2)
1(1)
3(2)
2
5(3)
5(3)

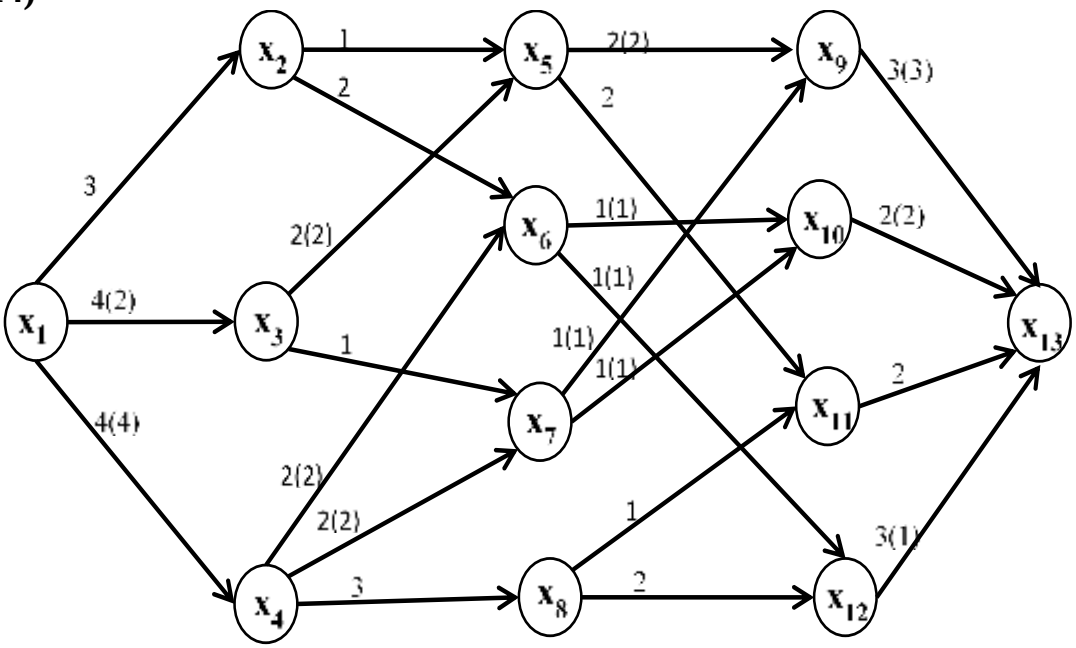
12)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
2(1)
4(1)
3(3)
3(3)
1(1)
1
1
3(1)
2(2)
1(1)
3
1(1)
2(2)
1
1
4(2)
3(1)
2(2)
6(4)
3(3)
4(3)

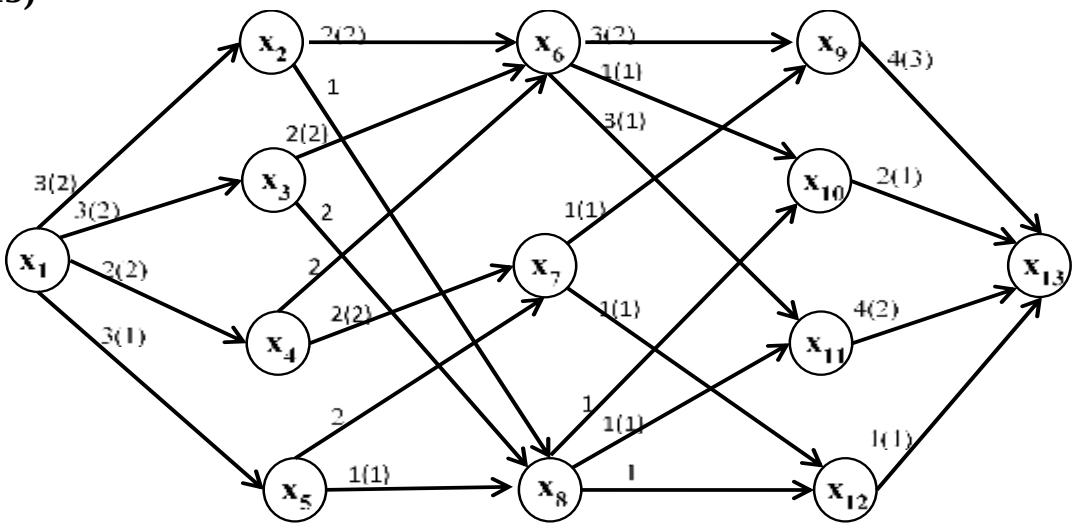
13)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
x15
3(3)
3(3)
4(2)
2(2)
1(1)
3(1)
2(2)
3
1
1
1
3(1)
1(1)
4(2)
1(1)
2
2(2)
1(1)
1
1(1)
1(1)
1
2(2)
6(4)
8(6)
1(1)
4(2)
3(1)

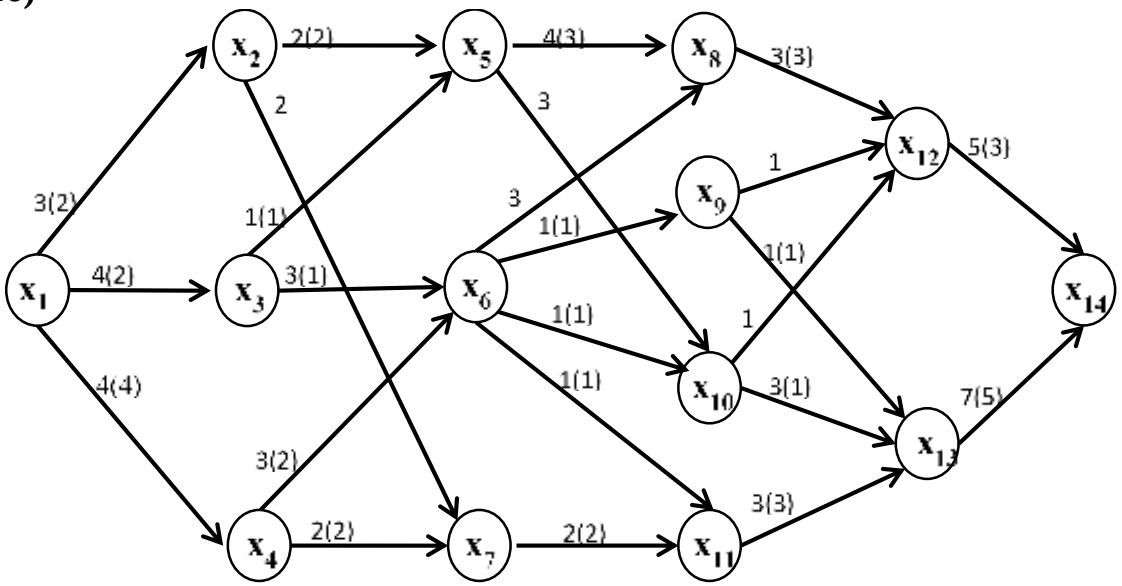
14)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3
4(2)
4(4)
1
2
2(2)
2(2)
1
3
2
2
1(1)
1(1)
1(1)
1(1)
1
2
3(3)
2(2)
3(1)

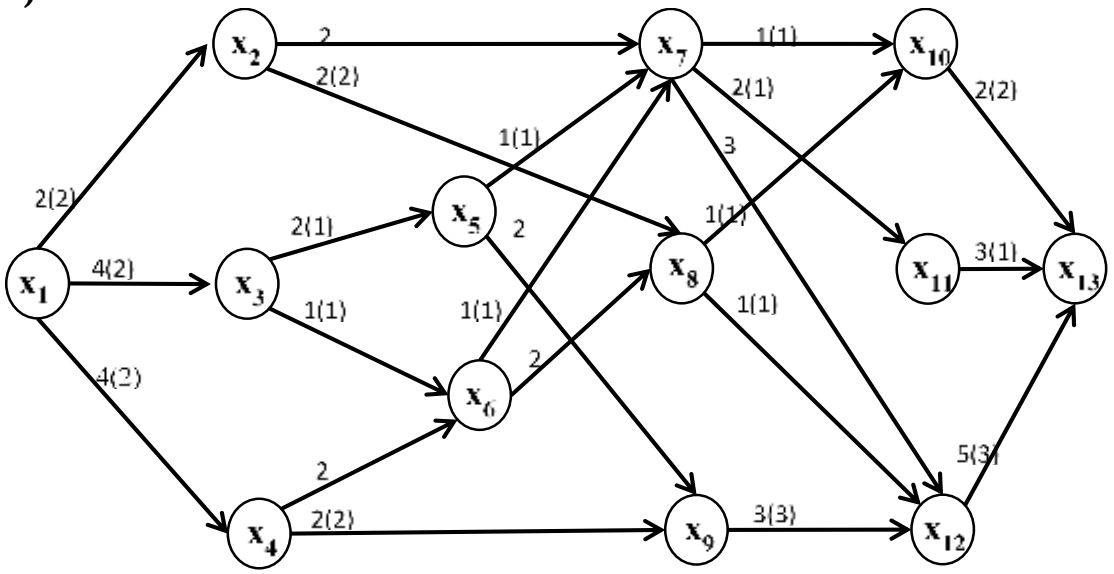
15)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(2)
3(2)
2(2)
3(1)
2(2)
1
2(2)
2
2
2
2
3(2)
1(1)
3(1)
1(1)
1
1(1)
1
1(1)
2(1)
4(3)
4(2)
1(1)

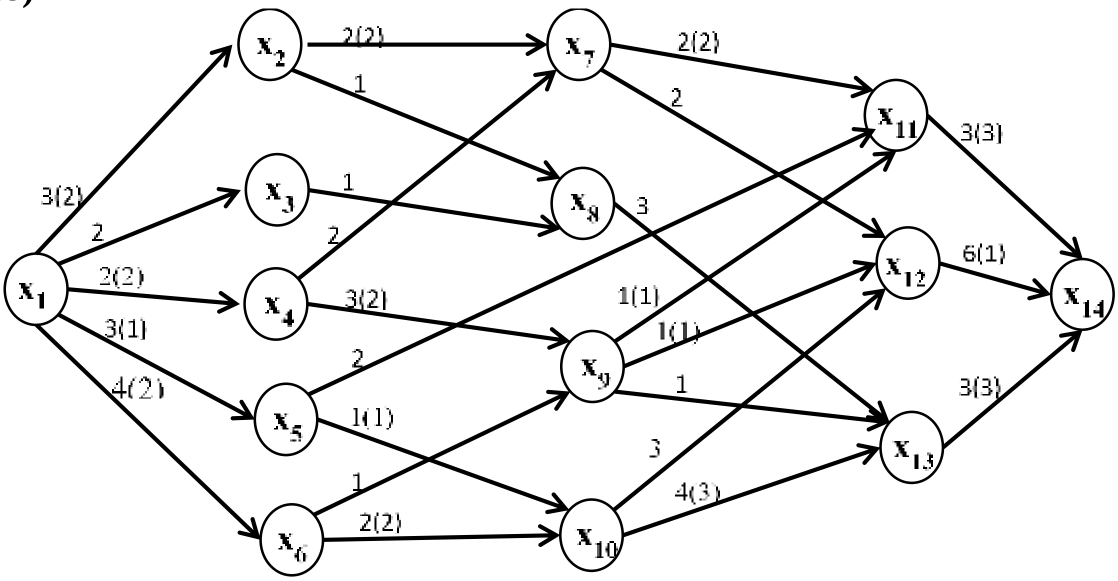
16)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
3(2)
4(2)
4(4)
2(2)
3(1)
2(2)
2
1(1)
3(2)
2(2)
4(3)
2
3
3
1(1)
1(1)
1(1)
1(1)
1
1
3(3)
1
1(1)
3(1)
2(2)
3(3)
5(3)
7(5)

17)

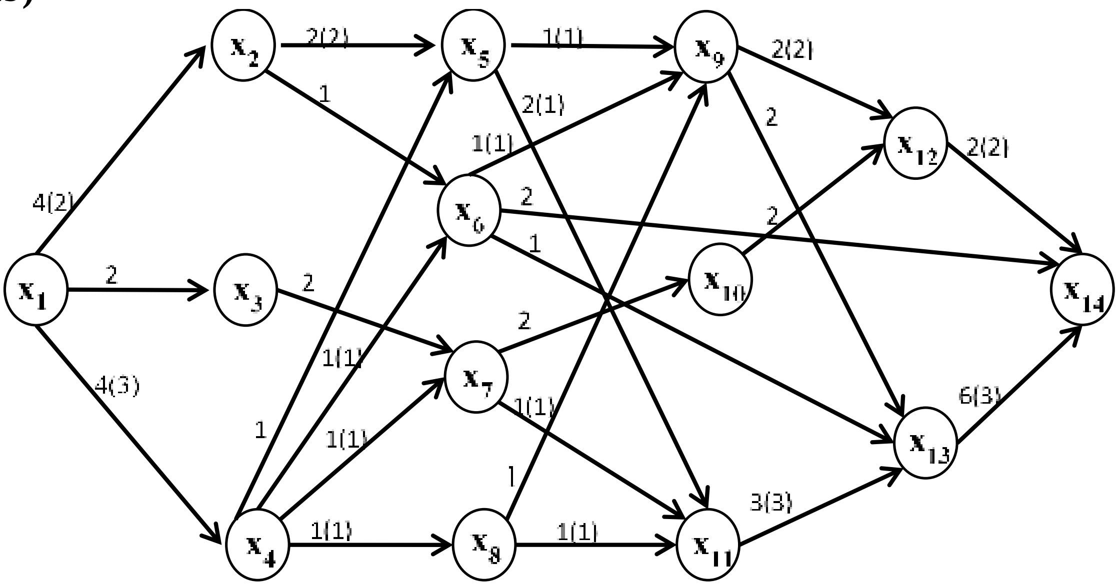

x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
2(2)
4(2)
4(2)
2(1)
1(1)
2
2(2)
2
2(2)
1(1)
1(1)
2
1(1)
2(1)
3
1(1)
1(1)
2
1(1)
3(3)
2(2)
1(1)
2(2)
3(1)
5(3)
3(3)

18)



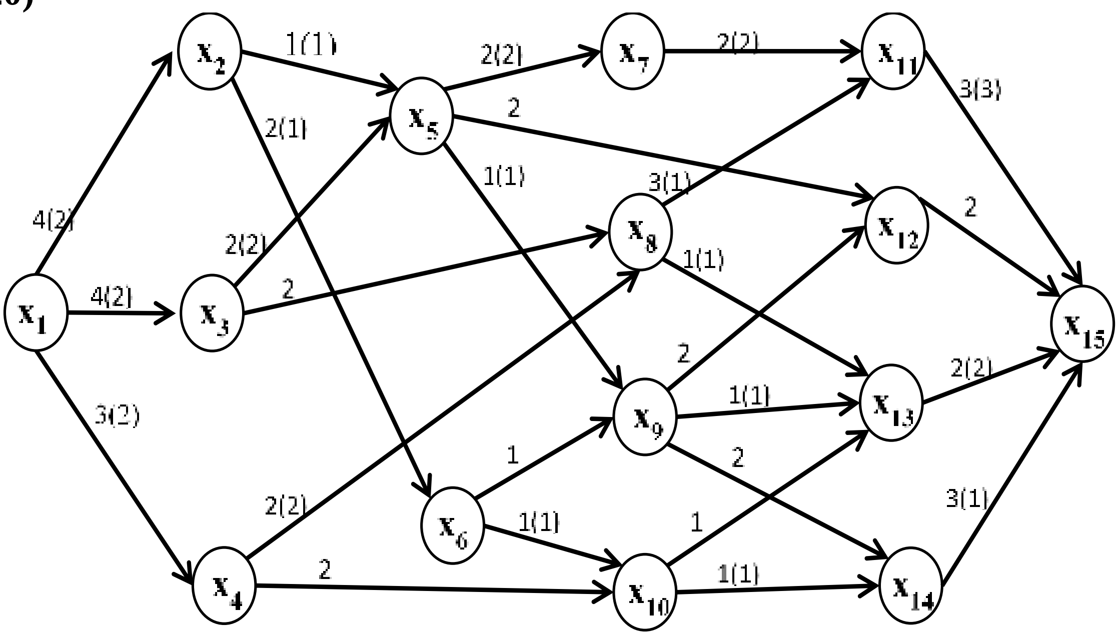
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
3(2)
2
2(2)
3(1)
4(2)
2(2)
1
1
2
3(2)
2
1(1)
1
2(2)
2(2)
2
3
1(1)
1(1)
1
3
4(3)
3(3)
6(1)
3(3)

19)



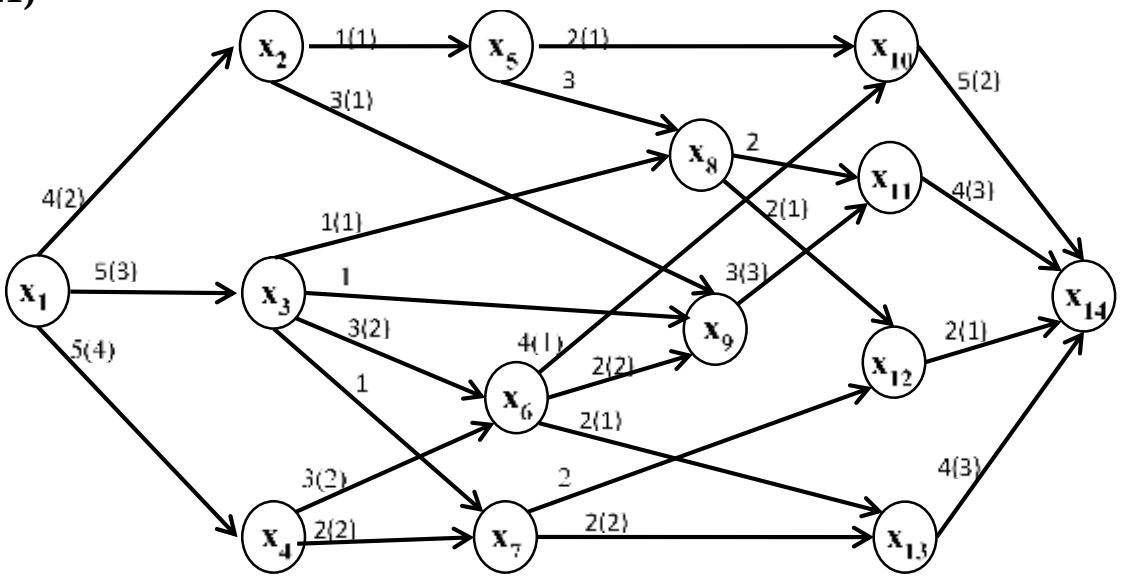
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
4(2)
2
4(3)
2(2)
1
2
1
1(1)
1(1)
1(1)
1(1)
1(1)
2(1)
1(1)
2
1
2
1(1)
1
1(1)
2(2)
2
2
3(3)
2(2)
6(3)

20)

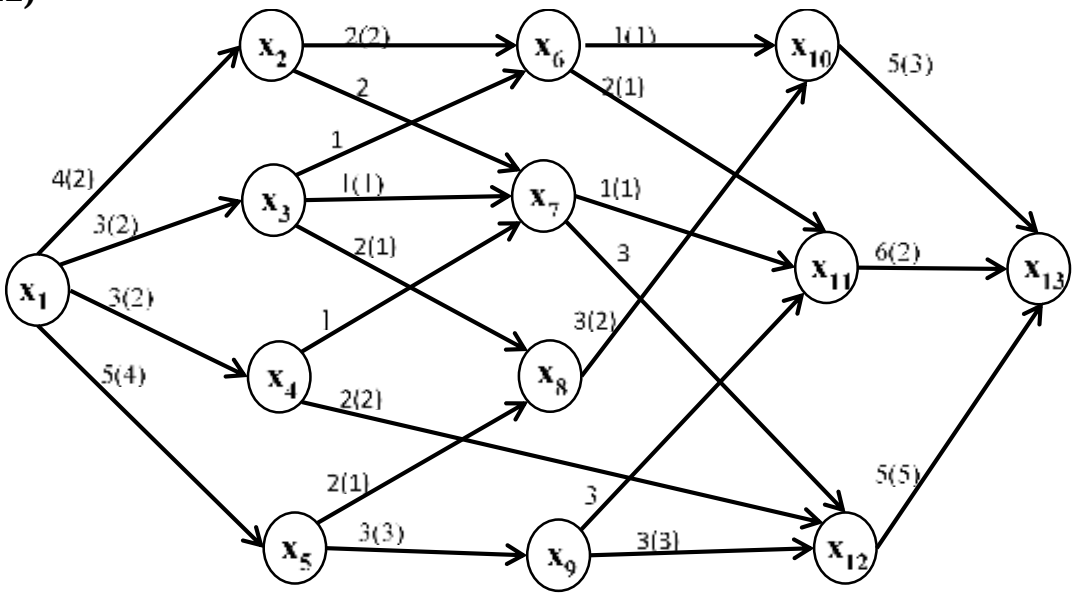


x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
x_{15}
4(2)
4(2)
3(2)
1(1)
2(1)
2(2)
2
2(2)
2
2(2)
2
1(1)
2(2)
3(1)
1(1)
2
1(1)
2
1
1(1)
1
1(1)
3(3)
2
2(2)
3(1)

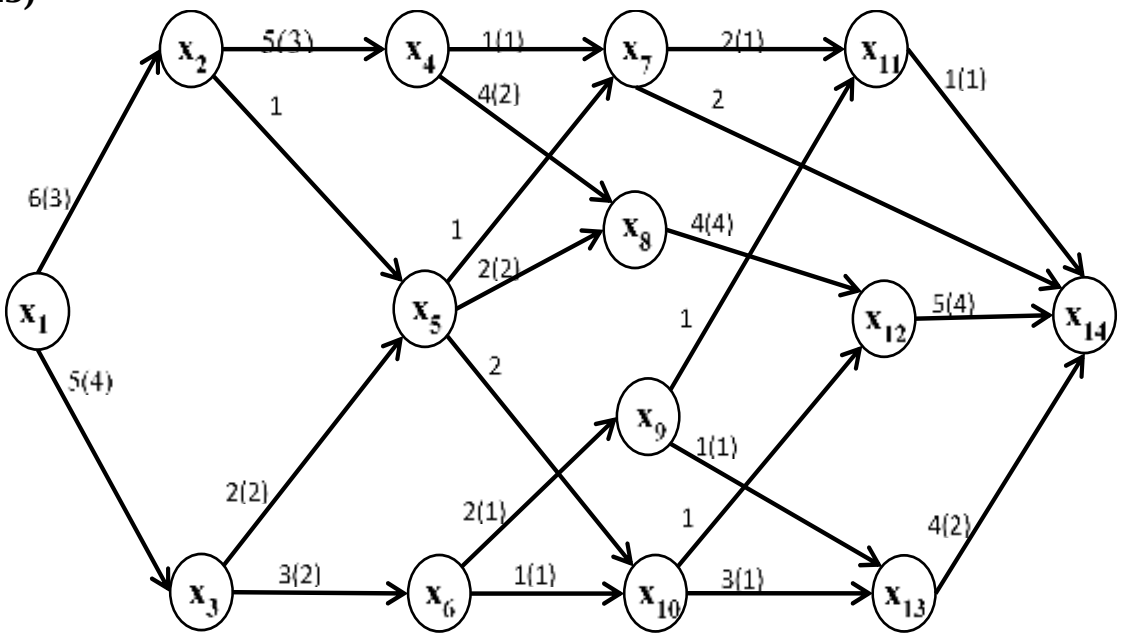
21)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
4(2)
5(3)
5(4)
1(1)
3(1)
1
3(2)
1
4(1)
2(2)
2(1)
3
2
3(3)
2(1)
2(2)
2(1)
5(2)
4(3)
2(1)
4(3)

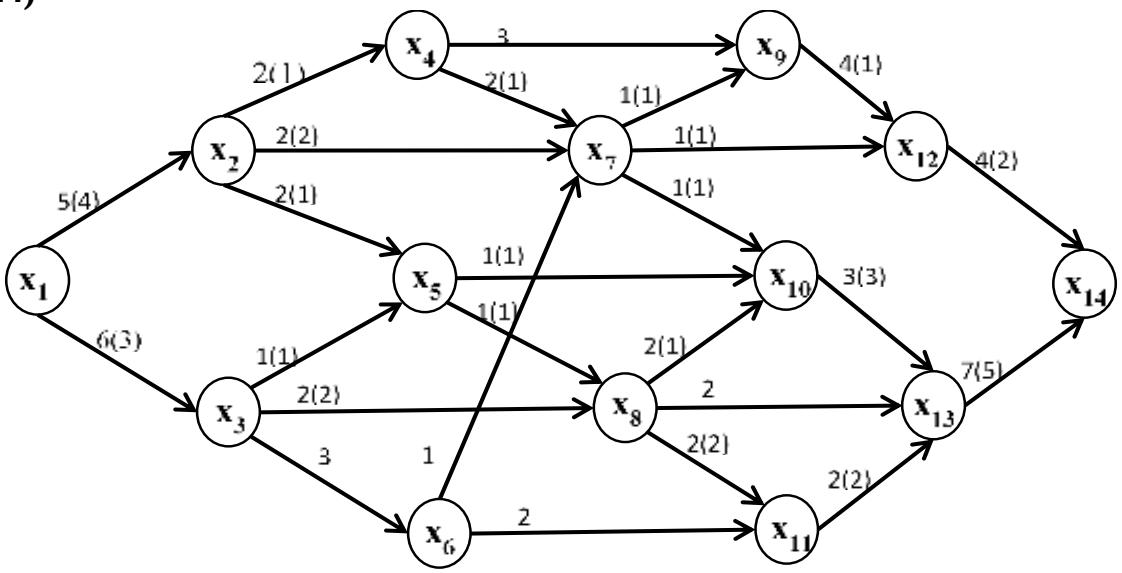
22)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
4(2)
3(2)
3(2)
5(4)
2(2)
2
1
1
2(1)
1
2(1)
3(3)
1(1)
1(1)
2(1)
3
3(2)
3
2(2)
3(3)
5(3)
6(2)
5(5)

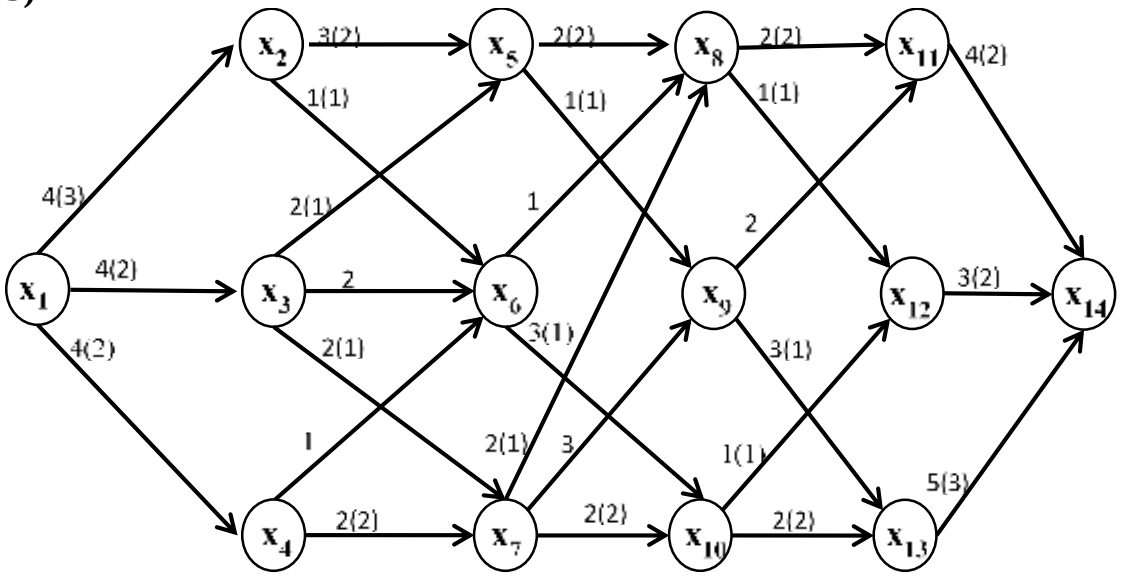
23)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
6(3)
5(4)
5(3)
1
4(2)
1
2(2)
2
2(2)
2(1)
3(2)
1(1)
1(1)
2
1
1
2
1
1(1)
3(1)
1(1)
4(4)
5(4)
4(2)

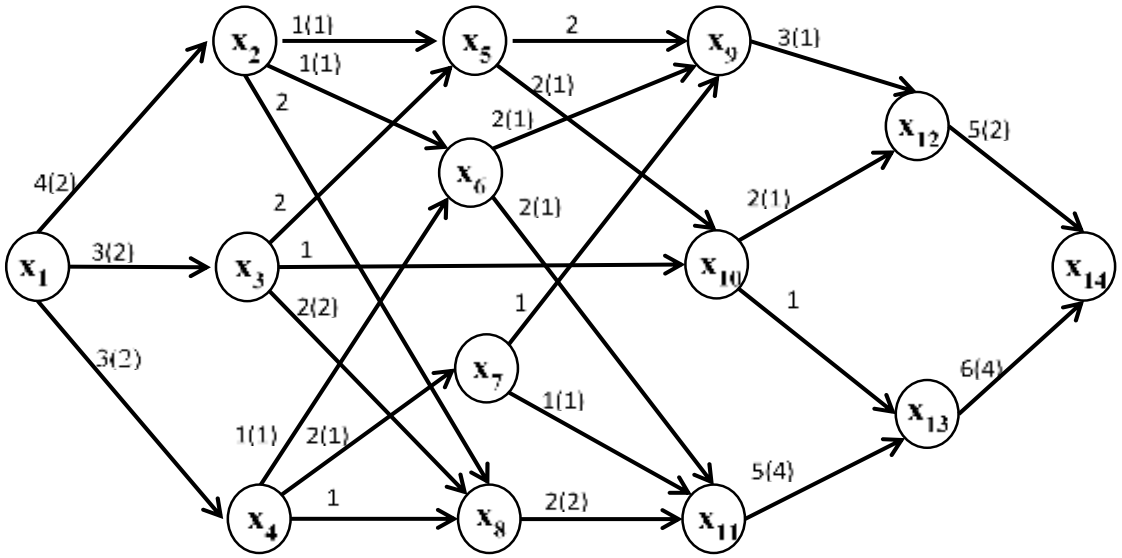
24)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
5(4)
6(3)
2(1)
2(2)
2(1)
3
2(1)
1(1)
1(1)
1(1)
1(1)
1(1)
2(1)
2(2)
3
1
2
2
3(3)
4(1)
4(2)
7(5)
2(2)
2(2)

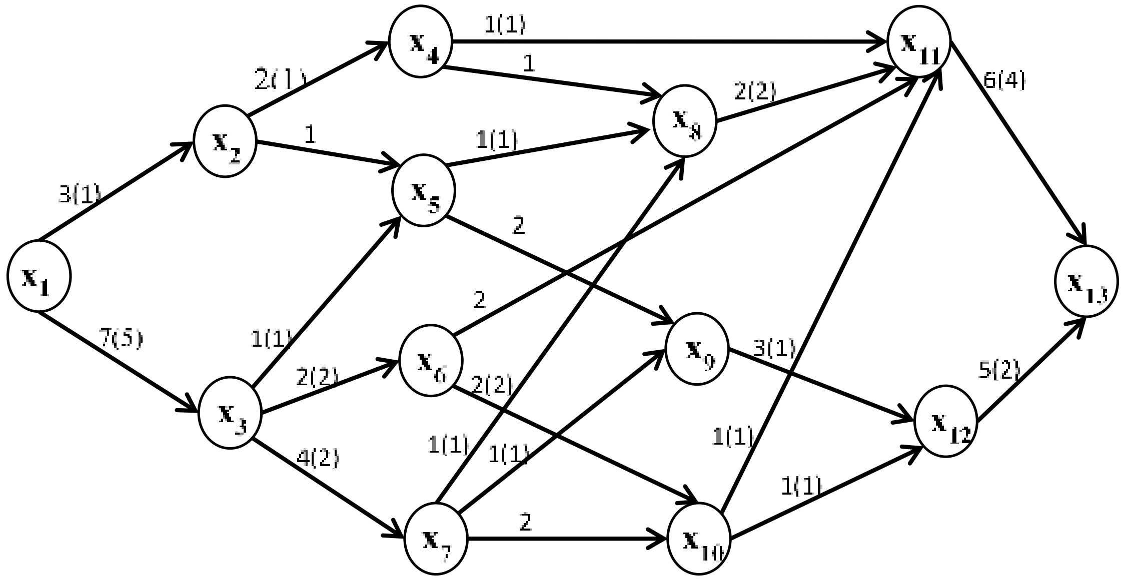
25)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
4(3)
4(2)
4(2)
3(2)
1(1)
2(1)
2
1
1
2(1)
2(1)
3
1(1)
1(1)
1(1)
2
3(1)
3(1)
2(2)
2(2)
2(2)
2(2)
4(2)
3(2)
5(3)

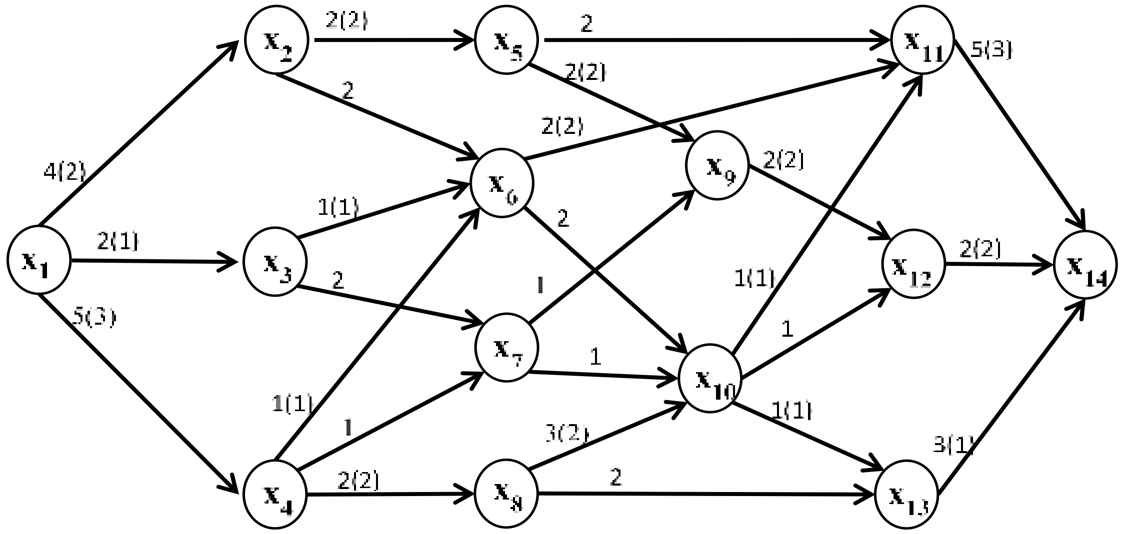
26)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
4(2)
3(2)
3(2)
1(1)
1(1)
2
2
2
2(2)
1(1)
2(1)
1
2(1)
2(1)
2(1)
1
2(1)
1
1
1
2
3(1)
5(2)
2(1)
1
6(4)
1(1)
5(4)
2(2)

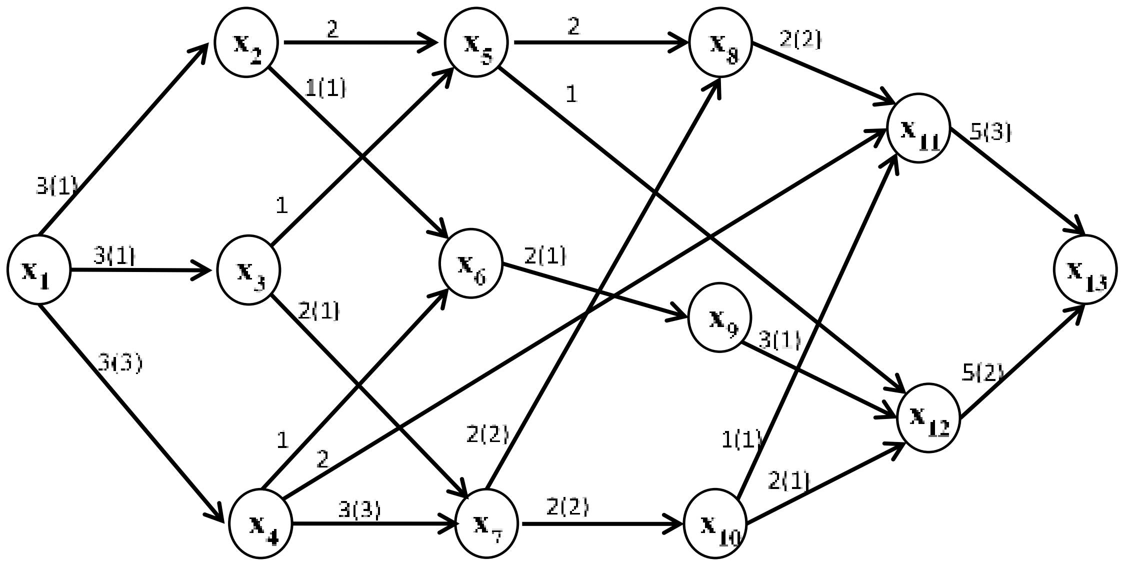
27)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(1)
7(5)
2(1)
1
1(1)
2(2)
4(2)
1(1)
1
1(1)
2
2
2(2)
1(1)
1(1)
2
2(2)
3(1)
1(1)
1(1)
6(4)
5(2)

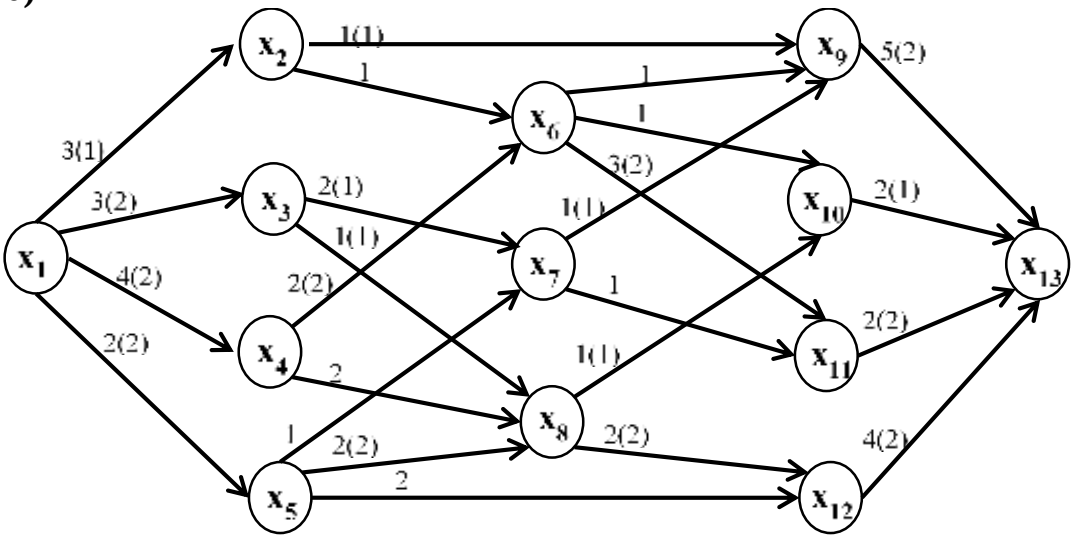
28)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
4(2)
2(1)
5(3)
2(2)
2
1(1)
2
1(1)
1
2(2)
2
2(2)
2(2)
2
1
1
3(2)
2
2(2)
1(1)
1
1(1)
5(3)
2(2)
3(1)

29)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(1)
3(1)
3(3)
2
1(1)
1
2(1)
1
2
3(3)
2
1
2(1)
2(2)
2(2)
2(2)
3(1)
1(1)
2(1)
5(3)
5(2)

30)


x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
3(1)
3(2)
4(2)
2(2)
1(1)
1
2(1)
1(1)
2(2)
2
1
2(2)
2
1
1
3(2)
1(1)
1
1(1)
2(2)
2(2)
5(2)
2(1)
2(2)
4(2)

Лабораторная работа №13
ПЛАНАРНЫЕ ГРАФЫ

Для графа (согласно индивидуального варианта задания):

1) определить хроматическое число;

2) построить плоское изображение графа или доказать, что граф непланарен.

Решение должно быть пошаговым с обязательной нумерацией каждого шага.

Индивидуальные задания по вариантам

1.

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

2.

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

3.

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

4.

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

5.

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

6.

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

7.

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

8.

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

9.

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

10.

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

11.

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

12.

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

13.

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

14.

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

15.

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

6.

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

17.

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

18.

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

19.

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

20.

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

21.

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

22.

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

23.

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

24.

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

25.

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

26.

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

27.

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

28.

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

29.

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

30.

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

Лабораторная работа №14
МАКСИМАЛЬНЫЙ ПОТОК В СЕТИ

Для орграфа (согласно индивидуального варианта задания) найти гамильтонов контур минимальной стоимости.

В решении должны присутствовать все промежуточные результаты.

Индивидуальные задания по вариантам

1)

	x_1	x_2	x_3	x_4	x_5
x_1	-	2	17	13	-
x_2	3	-	14	5	14
x_3	-	2	-	10	5
x_4	8	4	15	-	10
x_5	10	3	-	12	-

2)

	x_1	x_2	x_3	x_4	x_5
x_1	-	19	4	10	1
x_2	5	-	4	4	19
x_3	10	11	-	13	2
x_4	4	16	9	-	17
x_5	11	17	-	1	-

3)

	x_1	x_2	x_3	x_4	x_5
x_1	-	10	3	4	8
x_2	17	-	0	18	6
x_3	11	12	-	3	7
x_4	5	11	2	-	8
x_5	1	3	23	15	-

4)

	x_1	x_2	x_3	x_4	x_5
x_1	-	3	12	20	4
x_2	31	-	7	4	-
x_3	10	25	-	13	20
x_4	8	-	20	-	6
x_5	16	5	23	8	-

5)

	x_1	x_2	x_3	x_4	x_5
x_1	-	12	17	10	13
x_2	31	-	29	3	1
x_3	12	9	-	4	32
x_4	7	2	6	-	6
x_5	1	24	0	17	-

6)

	x_1	x_2	x_3	x_4	x_5
x_1	-	11	11	0	2
x_2	3	-	24	27	11
x_3	14	4	-	7	23
x_4	14	20	-	-	23
x_5	22	1	3	26	-

7)

	x_1	x_2	x_3	x_4	x_5
x_1	-	26	-	7	14
x_2	15	-	3	21	6
x_3	4	14	-	8	29
x_4	28	8	2	-	-
x_5	0	4	27	1	-

8)

	x_1	x_2	x_3	x_4	x_5
x_1	-	4	10	9	18
x_2	2	-	23	10	15
x_3	5	16	-	-	17
x_4	28	14	11	-	19
x_5	2	29	5	18	-

9)

	x_1	x_2	x_3	x_4	x_5
x_1	-	3	25	4	14
x_2	2	-	5	15	12
x_3	4	8	-	7	19
x_4	15	4	3	-	9
x_5	3	14	7	23	-

10)

	x_1	x_2	x_3	x_4	x_5
x_1	-	10	2	13	3
x_2	5	-	19	3	17
x_3	16	8	-	19	5
x_4	4	13	20	-	7
x_5	12	18	6	1	-

11)

	x_1	x_2	x_3	x_4	x_5
x_1	-	5	1	30	10
x_2	4	-	32	12	27
x_3	25	-	-	4	2
x_4	10	38	3	-	6
x_5	14	17	16	1	-

12)

	x_1	x_2	x_3	x_4	x_5
x_1	-	3	10	29	2
x_2	1	-	24	-	19
x_3	20	8	-	5	16
x_4	10	1	15	-	7
x_5	2	14	28	15	-

13)

	x_1	x_2	x_3	x_4	x_5
x_1	-	24	4	8	12
x_2	19	-	14	5	9
x_3	8	7	-	4	5
x_4	15	8	15	-	2
x_5	9	5	0	16	-

14)

	x_1	x_2	x_3	x_4	x_5
x_1	-	16	9	-	2
x_2	11	-	14	9	6
x_3	21	4	-	4	18
x_4	7	-	1	-	22
x_5	8	9	11	15	-

15)

	x_1	x_2	x_3	x_4	x_5
x_1	-	-	2	4	18
x_2	17	-	7	12	2
x_3	3	16	-	21	1
x_4	22	1	12	-	15
x_5	-	1	23	16	-

16)

	x_1	x_2	x_3	x_4	x_5
x_1	-	18	25	13	10
x_2	21	-	3	6	14
x_3	7	-	-	4	31
x_4	20	4	-	-	8
x_5	11	2	3	20	-

17)

	x_1	x_2	x_3	x_4	x_5
x_1	-	1	16	20	3
x_2	23	-	1	15	-
x_3	6	1	-	10	16
x_4	12	15	1	-	22
x_5	2	18	-	3	-

18)

	x_1	x_2	x_3	x_4	x_5
x_1	-	5	2	10	-
x_2	-	-	3	12	10
x_3	14	14	-	5	3
x_4	15	10	4	-	8
x_5	17	-	2	13	-

19)

	x_1	x_2	x_3	x_4	x_5
x_1	-	7	12	3	11
x_2	23	-	3	15	1
x_3	0	6	-	18	17
x_4	2	8	11	-	5
x_5	3	8	10	4	-

20)

	x_1	x_2	x_3	x_4	x_5
x_1	-	2	11	13	10
x_2	-	-	17	1	11
x_3	4	19	-	4	5
x_4	9	17	16	-	4
x_5	4	1	19	10	-

21)

	x_1	x_2	x_3	x_4	x_5
x_1	-	32	9	4	12
x_2	0	-	24	17	1
x_3	29	1	-	3	31
x_4	6	6	2	-	7
x_5	17	13	12	10	-

22)

	x_1	x_2	x_3	x_4	x_5
x_1	-	23	4	7	14
x_2	3	-	1	26	22
x_3	24	11	-	27	3
x_4	-	23	20	-	14
x_5	11	2	11	0	-

23)

	x_1	x_2	x_3	x_4	x_5
x_1	-	29	14	8	4
x_2	27	-	4	1	0
x_3	3	6	-	21	15
x_4	2	-	8	-	28
x_5	-	14	26	7	-

24)

	x_1	x_2	x_3	x_4	x_5
x_1	-	17	16	-	5
x_2	5	-	29	18	2
x_3	23	15	-	10	2
x_4	11	19	14	-	28
x_5	10	8	4	9	-

25)

	x_1	x_2	x_3	x_4	x_5
x_1	-	19	8	7	4
x_2	7	-	14	23	3
x_3	5	12	-	15	2
x_4	3	9	4	-	15
x_5	25	14	3	4	-

26)

	x_1	x_2	x_3	x_4	x_5
x_1	-	5	8	19	16
x_2	6	-	18	1	12
x_3	19	17	-	3	-
x_4	20	7	13	-	4
x_5	2	6	10	13	-

27)

	x_1	x_2	x_3	x_4	x_5
x_1	-	2	-	4	25
x_2	16	-	17	1	14
x_3	32	27	-	12	4
x_4	3	6	36	-	10
x_5	1	10	5	30	-

28)

	x_1	x_2	x_3	x_4	x_5
x_1	-	16	2	5	20
x_2	28	-	14	15	2
x_3	24	19	-	-	1
x_4	15	7	1	-	10
x_5	10	2	3	29	-

29)

	x_1	x_2	x_3	x_4	x_5
x_1	-	5	7	4	8
x_2	0	-	5	16	9
x_3	14	9	-	5	19
x_4	15	2	8	-	15
x_5	4	12	24	8	-

30)

	x_1	x_2	x_3	x_4	x_5
x_1	-	18	4	4	21
x_2	11	-	9	15	8
x_3	14	6	-	9	11
x_4	1	22	-	-	7
x_5	9	2	16	-	-

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бережной, В. В. Дискретная математика : учебное пособие / В. В. Бережной, А. В. Шапошников. — Ставрополь : СКФУ, 2016. — 199 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/155284 (дата обращения: 27.02.2023). — Режим доступа: для авториз. пользователей.
2. Веремчук, Н. С. Прикладная математика : учебно-методическое пособие / Н. С. Веремчук, Т. А. Полякова. — Омск : СибАДИ, 2022. — 198 с. — ISBN 978-5-00113-195-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/270887 (дата обращения: 27.02.2023). — Режим доступа: для авториз. пользователей.
3. Кравченко, Л. В. Прикладная математика: практикум : учебное пособие / Л. В. Кравченко, В. Н. Литвинов, В. В. Журба. — Ростов-на-Дону : Донской ГТУ, 2020. — 106 с. — ISBN 978-5-7890-1821-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/237986 (дата обращения: 27.02.2023). — Режим доступа: для авториз. пользователей.
4. Новоселов В. Г. Прикладная математика для системотехников. Дискретная математика в задачах и примерах. : учебное пособие. / В. Г. Новоселов, А. В. Скатков – Киев : УМК ВО, 1992. – 200 с.
5. Новиков Ф. А. Дискретная математика для программистов / Ф. А. Новиков. – СПб. : Питер, 2009. – 384 с.
6. Галушкина Ю. И. Конспект лекций по дискретной математике/ Ю.И. Галушкина, А. Н. Марьямов – М. : Айрис–пресс, 2007. – 176 с.
7. Белоусов А. Н. Дискретная математика / А. Н. Белоусов, Д. М. Грачев –М : МВТУ им, Баумана, 2001. - 744с.
8. Иванов Б. Н. Дискретная математика. Алгоритмы и программы/ Б. Н. Иванов – М : Лаборатория базовых знаний, 2001. - 288с.
9. Игошин В. И. Математическая логика и теория алгоритмов. — 2-е изд., стереотип.. — М. : Академия, 2008. — 448 с.
10. Мангушева И. П., Хрусталев П. М. Лекции по дискретной математике. Часть I. Элементы теории множеств и отношений. Функции алгебры логики. Функции k-значной логики : учебное пособие. – Саратов : Научная книга, 2010. - 68с.
11. Макоха А. Н., Сахнюк П. А., Червяков Н. И. Дискретная математика : учебное пособие. - М. : ФИЗМАТЛИТ, 2005 г. - 368 с.
12. Москинова Г. И. Дискретная математика. Математика для менеджера в примерах и упражнениях : учебное пособие, М. : Логос, 2000. - 240 с.
13. Набебин А. А., Тарасиков А. С. Алгебраическая спецификация программных систем М. : ИНЭК, 2012 г. - 84 с.
14. Перязев Н. А. Основы теории булевых функций / Перязев Н. А. - М. : Физматлит, 2000. - 109 с.

ДИСКРЕТНАЯ МАТЕМАТИКА И КОМПЬЮТЕРНАЯ ЛОГИКА. ТЕОРИЯ МНОЖЕСТВ, ТЕОРИЯ ОТНОШЕНИЙ, ТЕОРИЯ ГРАФОВ

Методические указания

Составители : **Ченгарь** Ольга Васильевна,
Малицкая Александра Александровна

В авторской редакции

Изд. № 83/2023. Объем 10,25 п.л. Усл. печ. л. 9,53. Уч.-изд. л. 10,045.
РИИЦМ ФГАОУ ВО «Севастопольский государственный университет»