МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«НАЦІОНАЛЬНИЙ ГІРНИЧИЙ УНІВЕРСИТЕТ»

І.В. Новицький
С.А. Ус

ДИСКРЕТНА МАТЕМАТИКА В ПРИКЛАДАХ І ЗАДАЧАХ

Навчальний посібник

Дніпропетровськ
НГУ
2013

УДК 519.17:519.45(075.8)
ББК 22.176я73
Н73

Рекомендовано редакційною радою НГУ як навчальний посібник з дисципліни «Дискретна математика» для студентів напряму підготовки 6.040303 Системний аналіз (протокол № 5 від 17.05.2012).

Рецензенти:
Д.Г. Зєлєнцов, д-р техн. наук, професор (Український державний хіміко-технологічний університет, завідувач кафедри інформаційних систем);
Л.С. Коряшкіна, канд. фіз.-мат. наук, доцент (Дніпропетровський національний університет, кафедра обчислювальної математики і математичної кібернетики).

Новицький І.В.
Н73 Дискретна математика: навч. посібник / І.В. Новицький, С.А. Ус. – Д. : Національний гірничий університет, 2013. – 89 с.

Навчальний посібник охоплює матеріал, передбачений програмою дисципліни “Дискретна математика” для студентів напряму підготовки 6.040303 Системний аналіз. Розглянуто базові теоретичні засади дискретної математики, описано методи й алгоритми, необхідні для розв’язування задач, показано їх застосування на конкретних прикладах.

Навчальний посібник має на меті допомогти студентам-заочникам у самостійному вивченні нормативної дисципліни «Дискретна математика» під час виконання контрольних робіт і підготовки до практичних і лабораторних занять.

Книгу розраховано на осіб, які знають математику в межах вузівського курсу, й рекомендовано для студентів технічних спеціальностей і тих, хто використовує методи дискретної математики при розв’язуванні практичних задач.

УДК 519.17:519.45(075.8)
ББК 22.176я73

ЗМІСТ

ВСТУП

Дискретна математика являє собою сукупність математичних дисциплін, які вивчають властивості абстрактних переривчастих об'єктів, тобто ознаки математичних моделей об'єктів, процесів, залежностей, наявних у реальному світі, якими оперують у різних галузях знань.

Дискретна математика, або дискретний аналіз – це самостійний розділ сучасної математики, що вивчає властивості різних структур, які мають скінченний характер. Вони можуть виникати як у самій математиці, так і в її застосуваннях. До таких структур відносять об'єкти, що мають переривчастий (дискретний) характер, на відміну від неперервних об'єктів, досліджуваних класичною математикою.

Поділ математики на дискретну й класичну досить умовний. Наприклад, апарат теорії множин і теорії графів використовується при вивченні не тільки дискретних, але й неперервних об'єктів. З іншого боку, сама дискретна математика використовує засоби, розроблені в класичній математиці. Разом із тим, характер об'єктів, досліджуваних дискретною математикою, настільки своєрідний, що використання тільки методів класичної математики часто буває недостатнім для їхнього вивчення.

Стимулом до розвитку багатьох напрямів дискретної математики стали потреби теоретичної кібернетики, безпосередньо пов'язаної з розвитком ЕОМ. Застосування ЕОМ з метою комплексної автоматизації інформаційної діяльності принципово змінило характер контакту людини з машиною. Якщо раніше комп'ютер опановували тільки ті, хто безпосередньо його обслуговував, електронники, оператори, то в сучасному світі без машинної обробки інформації не обійдеться жодна галузь діяльності.

Дискретна математика – порівняно новий науковий напрям, що поєднує окремі розділи математики, раніше сформовані як окремі теорії. Основні з них – теорія множин, теорія графів, математична логіка, теорія автоматів. Розділи з відповідними назвами якраз і становлять зміст даного посібника.

РОЗДІЛ 1
ОСНОВИ ТЕОРІЇ МНОЖИН

У розділі висвітлено зміст основних понять теорії множин і показано їх практичне застосування при виконанні операцій над множинами.

§ 1.1. Основні поняття

Засновник теорії множин Г. Кантор дав таке визначення її базового поняття: *множина* – це сукупність елементів, що розглядаються як єдине ціле. Задати множину можна або простим переліком її елементів, або назвавши ознаку, яка властива всім її елементам. Наприклад:

$$X = \{x_1, x_2, x_3, x_4\}, \qquad Y = \{y \mid y - \textit{ціле число}\}.$$

Тут множину X задано переліком елементів, а множину Y – зазначенням властивості елементів.

Запис $x_2 \in X$ означає, що елемент x_2 належить множині X. Символом $\varnothing$ позначається *порожня множина*, яка не містить жодного елемента.

Будь-який набір елементів, кожний з яких належить множині X, називається *підмножиною* множини X.

Наприклад, нехай $X = \{1,2,3,4,5\}$; $A = \{1,3,5\}$, тоді множина A являє собою підмножину множини X.

Запис: $A \subset X$ означає, що підмножина A міститься в множині X. За визначенням кожна множина є підмножиною самої себе, тобто $X \subset X$. Крім того, $\varnothing \subset X$. Таким чином, кожна підмножина має принаймні дві підмножини: порожню множину і саму себе.

Якщо множина X складається з n елементів, то завжди існує 2^n її підмножин. Наприклад, коли множина $X = \{1,2,3\}$, тобто $n = 3$, то існує така кількість її підмножин: $2^3 = 8$, а саме: $\{\varnothing\}$; $\{1\}$; $\{2\}$; $\{3\}$; $\{1,2\}$; $\{1,3\}$; $\{2,3\}$; $\{1,2,3\}$.

Множина, яка містить усі елементи, розглянуті в дискретній математиці, називається *універсальною множиною* S. Очевидно, що будь-яка множина являє собою підмножину універсальної множини S, тобто $X \subset S$.

Для наочного зображення співвідношень між множинами використовуються *діаграми Ейлера*. Вони зображують універсальну множину у вигляді прямокутника, усередині якого розміщуються інші множини, їх зображують у вигляді кіл (див. рис. 1.1).

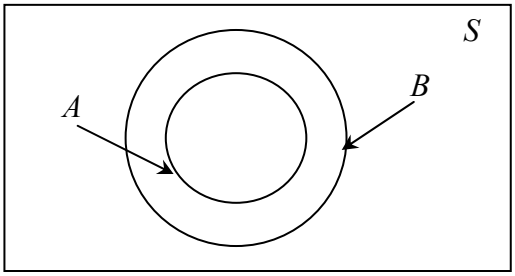


Рис. 1.1. Приклад діаграми Ейлера

Нагадаємо загальноприйняті позначення числових множин, які відіграють важливу роль у теорії множин:

$N = \{1,2,3,4...\}$ – множина натуральних чисел;

$Z\{...-3,-2,-1,0,1,2,3...\}$ – множина цілих чисел;

$Q = \left\{ \dfrac{g_1}{g_2} \middle| g_1 \in Z,\, g_2 \in N \right\}$ – множина раціональних чисел;

$R = \{x | a \le x \le b\}$ – множина дійсних чисел відрізка $(a;b)$, часто $a = 0$; $b = 1$.

Очевидно, що $N \subset Z \subset Q$.

Множина, що містить скінченну кількість елементів, називається *скінченною*. Множина, що не є скінченною, називається *нескінченною*.

Будь-яка множина X має важливу універсальну характеристику – *потужність* множини (кардинальне число). Вона позначається таким чином: $|X|$.

Потужність скінченних множин дорівнює кількості елементів множини. Наприклад: множина $A = \{a_1, a_2, a_3, a_4, a_5\}$, тоді $|A| = 5$.

Кардинальні числа мають місце також і для нескінченних множин. Так, Г. Кантор запропонував позначити потужність множини натуральних чисел символом α_0 (читається як *алеф-нуль*), тобто $|N| = \alpha_0$.

Г. Кантор сформулював також принцип порівняння потужностей двох множин: *якщо між елементами двох множин можна встановити взаємно однозначну відповідність, то ці множини рівнопотужні*. Наприклад, нехай дано множини: $A = \{1,5,7\}$ та $B = \{b_1, b_2, b_3\}$. Взаємно однозначну відповідність між ними встановлено таким чином:

$$\begin{array}{c} A = \{1,\ 5,\ 7\} \\ \updownarrow\ \updownarrow\ \updownarrow \\ B = \{b_1, b_2, b_3\}. \end{array}$$

Зрозуміло, що $|A| = |B| = 3$, тобто ці множини рівнопотужні.

Сформульований принцип можна також застосовувати для порівняння потужностей нескінченних множин.

Наприклад: $G=\{2,4,6,8...\}$, це множина парних натуральних чисел. Очевидно, що $G\subset N$, однак, ми можемо встановити відповідність між цими множинами таким чином:

$$\begin{array}{c} 1\ 2\ 3\ 4... \\ \updownarrow\updownarrow\updownarrow\updownarrow \\ 2\ 4\ 6\ 8... \end{array}$$

а це означає, що $|G|=|N|$.

Аналогічно можна показати, що $|N|=|Z|=|Q|$.

Усі множини, які рівнопотужні множині натуральних чисел *N*, називаються *лічильними* й мають потужність α_0.

Існують нескінченні множини, які не можна поставити у відповідність натуральному ряду чисел, наприклад, множина дійсних чисел *R*. Такі множини зумовлені поняттям «неперервність» і мають потужність континууму (позначається *c*) й називаються *континуальними*.

§ 1.2. Операції із множинами

1. *Рівність*

Дві множини *A* і *B* дорівнюють одна одній тоді й тільки тоді, коли кожен елемент множини *A* є елементом множини *B* і навпаки. Тобто $A=B$, якщо $A\subset B$ й $B\subset A$.

2. *Об'єднання*

Об'єднанням або *сумою* двох множин *A* і *B* (позначається $A\cup B$) є множина, які містить усі елементи, що належать принаймні одній із множин *A* чи *B*. Наприклад, якщо множина $A=\{2,5,8\}$, $B=\{0,5,6,7\}$, то їх об'єднання $A\cup B=\{0,2,5,6,7,8\}$.

Зображення об'єднання множин за допомогою діаграм Ейлера бачимо на рис.1.2, де штрихуванням показано результат об'єднання множин *A* та *B*.

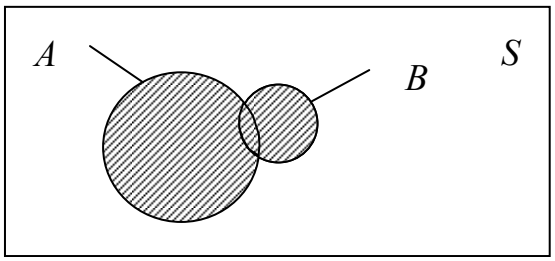


Рис.1.2. Графічне подання об'єднання множин за допомогою діаграм Ейлера

Відносно операції об'єднання будуть справедливими такі закони:

асоціативний: $(A \cup B) \cup C = A \cup (B \cup C)$;

комутативний: $A \cup B = B \cup A$.

Крім того, для будь-яких множин A та B будуть справедливими співвідношення:

$A \cup A = A$; $A \cup \varnothing = A$; $A \cup S = S$; $A \cup B = B$, якщо $A \subset B$.

3. *Перетин*

Перетином або *добутком* двох множин A і B (позначається $A \cap B$) називають сукупність елементів, які належать множинам A і B одночасно. Наприклад, перетином множин: $A = \{2, 5, 8\}$ і $B = \{0, 5, 6, 7\}$, буде така множина: $A \cap B = \{5\}$.

За допомогою діаграм Ейлера операцію перетину множин можна проілюструвати таким чином:

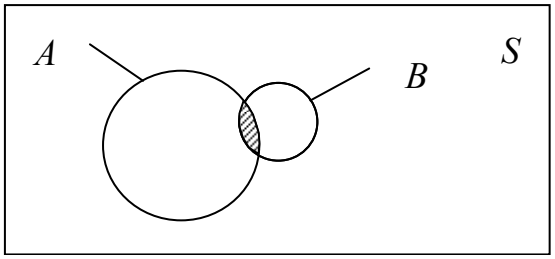


Рис.1.3. Графічне подання перетину множин A і B

Для цієї операції будуть справедливими такі закони:

асоціативний: $(A \cap B) \cap C = A \cap (B \cap C)$;

комутативний: $A \cap B = B \cap A$.

Крім того, для всіляких множин A та B будуть правильними такі співвідношення:

$A \cap A = A$; $A \cap \varnothing = \varnothing$; $A \cap S = A$; $A \cap B = A$, *якщо* $A \subset B$.

Операції об'єднання й перетину пов’язані також дистрибутивними законами:

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

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

У цьому неважко переконатися за допомогою діаграм Ейлера (див. рис.1.4).

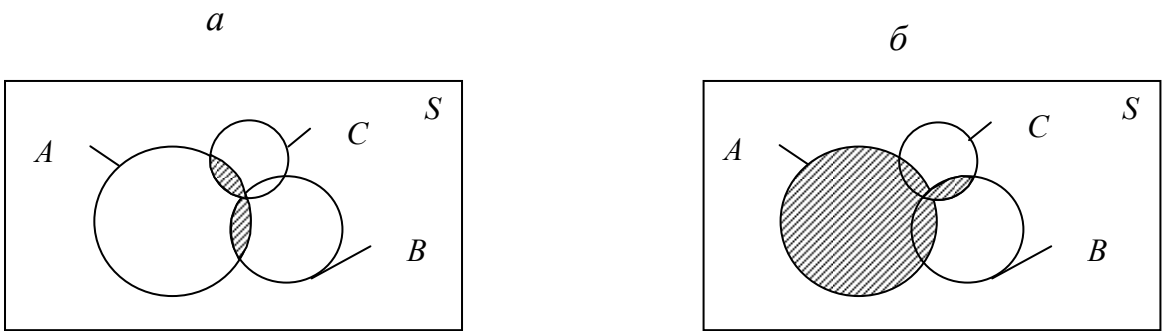


Рис. 1.4. Графічне подання дистрибутивних законів за допомогою діаграм Ейлера: *а* – $A\cap(B\cup C)=(A\cap B)\cup(A\cap C)$, *б* – $A\cup(B\cap C)=(A\cup B)\cap(A\cup C)$.

4. *Доповнення*

Доповненням множини A до універсальної множини S називається множина $\overline{A}$, яка включає в себе всі елементи універсальної множини S, крім тих, що належать множині A. На рис. 1.5. подано зображення доповнення множини A за допомогою діаграм Ейлера.

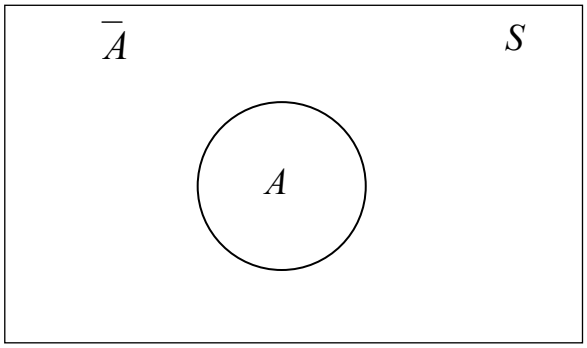


Рис.1.5. Графічна інтерпретація доповнення множини A

Відносно операції доповнення будуть справедливими такі твердження:

$$\overline{\varnothing}=S;\ \overline{S}=\varnothing;\ \overline{\overline{A}}=A;$$
$$A\cup\overline{A}=S;\ A\cap\overline{A}=\varnothing.$$

Крім того, із поняттям доповнення пов'язані закони де Моргана, а саме:

$$\left(\overline{A\cup B}\right)=\overline{A}\cap\overline{B};$$
$$\left(\overline{A\cap B}\right)=\overline{A}\cup\overline{B}.$$

Вони також можуть бути легко доведені за допомогою діаграм Ейлера.

5. *Різниця*

Різницею двох множин A та B (позначається як $A-B$ або A/B) називається множина, що складається з тих елементів множини A, які не містяться в множині B. На рис.1.6 її показано штриховкою.

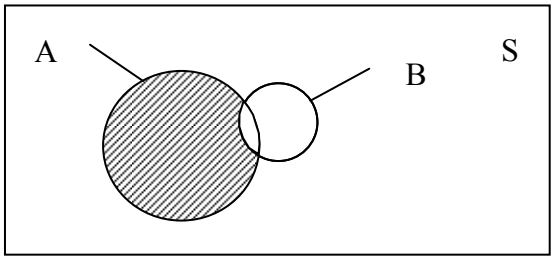


Рис.1.7. Графічна інтерпретація різниці множин A та B

Приклад. Нехай множини: $A = \{2, 5, 8\}$, $B = \{0, 5, 6, 7\}$, тоді їх різницею буде така множина: $A - B = \{2, 8\}$.

Відносно різниці множин будуть справедливими такі твердження:

$$A - A = \varnothing;$$
$$A - \varnothing = A;$$
$$A - S = \varnothing;$$
$$S - A = \overline{A}.$$

§ 1.3. Приклади розв'язування задач

1. Нехай задано множини: $A = \{3, 6, 7, 9\}$ та $B = \{1, 6, 8, 9\}$. Знайти їх об'єднання, перетин і різниці $A - B$ й $B - A$.

Розв'язування

Об'єднання множин включає елементи, які належать принаймні одній із множин, тому $A \cup B = \{1, 3, 6, 7, 8, 9\}$.

Перетин включає лише спільні елементи цих множин, отже, $A \cap B = \{6, 9\}$.

Тепер знайдемо різницю $A - B$. Вона включає елементи множини A, які не належать множині B, тобто $A - B = \{3, 7\}$.

Різниця $B - A$ включає елементи множини B, які не належать множині A, отже, $B - A = \{1, 8\}$.

2. Нехай $A = N; B = Z; (A \subset B)$. Знайти їхнє об'єднання, перетин та різниці $A - B$ і $B - A$.

Розв'язування:

Оскільки $A \subset B$, то $A \cup B = B$, $A \cap B = A$ і $A - B = \varnothing$.

За визначенням $B - A = \{0, -1, -2, -3...\}$.

3. Задано множини: $A = \{x | x \in R; 2 < x \le 5\}$; $B = \{x | x \in R; 0 \le x < 3\}$. Знайти їхнє об'єднання, перетин і різниці $A - B$ та $B - A$.

Розв'язування

$A \cup B = \{x | x \in R; 0 \le x \le 5\}$,

$A \cap B = \{x | x \in R; 2 < x < 3\}$,

$A - B = \{x | x \in R; 3 \le x \le 5\}$,

$B - A = \{x | x \in R; 0 \le x \le 2\}$.

4. Знайти множини A й B, якщо їх різниці такі: $A - B = \{a\}$, $B - A = \{b, e\}$, а $A \cup B = \{a, b, c, d, e\}$.

Розв'язування

Елементи множин A й B знайдемо за допомогою діаграм Ейлера (див. рис.1.8). Елемент a міститься тільки в множині A, елементи b, e – лише в множині B, тоді (враховуючи останню умову) елементи c, d належать перетину множин A та B.

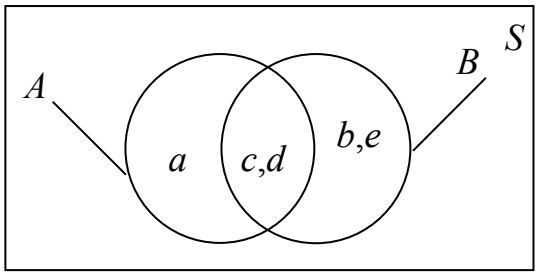


Рис. 1.8. Графічна інтерпретація задачі 4

Отже, $A = \{a, c, d\}$; $B = \{b, c, d, e\}$.

5. Виразити за допомогою множин A, B й C заштриховану ділянку D на діаграмі (рис. 1.9).

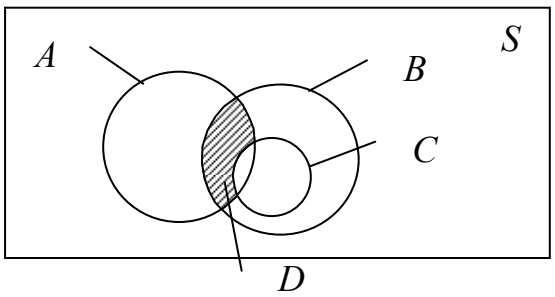


Рис.1.9. Графічне подання умови задачі 5

Розв'язування

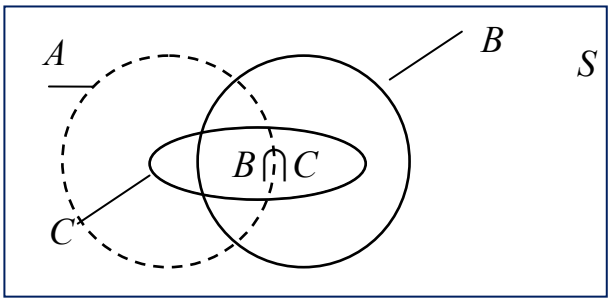
Згідно з діаграмою Ейлера заштрихована ділянка відповідає такому виразу: $D=\left(A\bigcap B\right)-C=\left(B-C\right)\bigcap A=\left(A-C\right)\bigcap B$.

6. Заштрихувати на діаграмі Ейлера ділянку, відповідну такій множині:

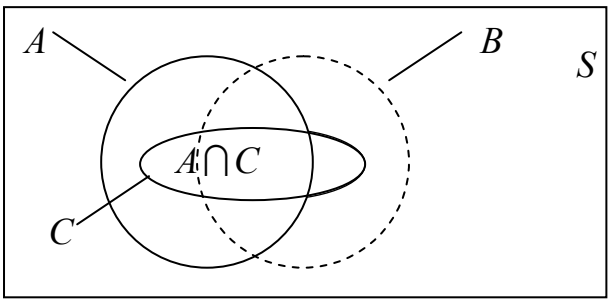
$$D=\left(B\bigcap C\right)-\left(A\bigcap C\right).$$

Розв'язування

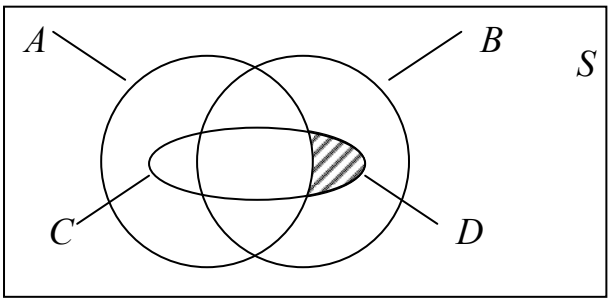
Спочатку зобразимо на діаграмі множини B, C та їхній перетин (рис.1.10, *а*). Потім перетин множин A й C (рис.1.10, *б*) і нарешті, розв'язок задачі подано на рис. 1.10, *в*.



а



б



в

Рис.1.10. Графічне подання етапів розв'язування задачі 7

7. Уведемо такі позначення: A – множина працівників заводу; B – множина працівників-чоловіків того самого заводу. Що означають множини $A-B$ і $\left(A\bigcap B\right)-B$?

Розв'язування

$A-B$ являє собою множину працівників заводу, не враховуючи робітників-чоловіків, тобто $A-B$ це множина працівників-жінок цього заводу. Оскільки множина $A\bigcap B=B$ (враховуючи, що $B\subset A$), то $\left(A\bigcap B\right)-B$ являє собою порожню множину.

8. Задано множини: $A=\left\{x\middle|x\in R;-5<x<5\right\}$, $B=Z$,$C=N$. Знайти такі множини: $D=A\bigcap\left(B-C\right)$, $E=A\bigcap\left(C-B\right)$, $F=\left(C-A\right)\bigcap B$, $G=A\bigcap B\bigcap C$.

Розв'язування

Множина $B-C$ являє собою множину цілих від'ємних чисел, тому множина D включає цілі від'ємні числа, які належать інтервалу (–5; 5), тобто $D=\left\{-4,-3,-2,-1,0\right\}$.

Оскільки $C\subset B$, то $C-B=\varnothing$, і, відповідно $E=\varnothing$.

Множина $C-A$ складається з натуральних чисел, більших п'яти, і, враховуючи, що $C\subset B$, маємо такий результат $F=\left\{5,6,7,8...\right\}$.

Аналогічно: $G=\left\{1,2,3,4\right\}$.

9. Знайти множини A і B, якщо: *а*) $A\bigcap B=\left\{3,5\right\}$; $B\subset A$; $A-B=\left\{7\right\}$; *б*) $A\bigcup B=\left\{2,5,7,8,10\right\}$; $A-B=\varnothing$; $B-A=\left\{7,10\right\}$.

Розв'язування

а) Оскільки $B\subset A$, то $A\bigcap B=B$, тому $B=\left\{3,5\right\}$. З огляду на це, множина A включає всі елементи множини B й елементи множини $A-B$, таким чином, $A=\left\{3,5,7\right\}$.

б) Враховуючи, що $A-B=\varnothing$, можемо зробити висновок: $A\subset B$, тоді $A\bigcup B=B$ й $B=\left\{2,5,7,8,10\right\}$. Множина $B-A$ містить елементи B, які не входять в A, тому $A=\left\{2,5,8\right\}$.

10. Задано множини: $A=\left\{1,3,5,7,9\right\}$, $B=\left\{1,2,5,7\right\}$, $C=\left\{4,7\right\}$. Виразити через A, B, C множини: $D=\left\{1,5\right\}$; $E=\left\{2,4\right\}$.

Розв'язування

$D=\left(A-C\right)\bigcap B,$

$E=\left(B\bigcup C\right)-A.$

Питання для самоконтролю

1. Дайте визначення поняття «множина».
2. Якими способами можна задати множину?
3. Скільки існує підмножин скінченної множини, складеної з n елементів?

4. Що являють собою діаграми Ейлера? З якою метою їх використовують?

5. Чому дорівнює потужність скінченної множини?

6. У якому випадку дві нескінченні множини будуть рівнопотужними?

7. Які множини називаються лічильними?

8. Назвіть основні операції над множинами.

9. Що являє собою різниця двох множин? Їх об'єднання? Перетин?

10. Дайте визначення доповнення множини.

11. Сформулюйте закони, яким підпорядковуються операції над множинами.

12. Які властивості характерні для кожної з операцій над множинами?

13. Чи виконується в загальному випадку комутативний закон для операції «різниця множин»?

Задачі для самостійного розв'язування

1. Задано множини: $A=\{0,3,5,8\}$ і $B=\{0,4,5,9\}$. Знайти їх об'єднання, перетин і різниці.

2. Дано множини: $A=N$, $B=\{x|x-$*натуральне парне*$\}$. Знайти для цих множин об'єднання, перетин і різниці $A-B$ та $B-A$.

3. Маючи такі множини: $A=\{x|x\in R;-1\le x<4\}$, $B=\{x|x\in R;0<x\le 6\}$, визначити їх об'єднання, перетин і різниці $A-B$ та $B-A$.

4. Знайти множини *A* і *B*, якщо:

$$A-B=\{a,b\};$$
$$B-A=\{d\};$$
$$A\cap B=\{c\}.$$

5. Виразити через множини *A*, *B* і *C* заштриховану ділянку *D* на діаграмі Ейлера (рис.1.11).

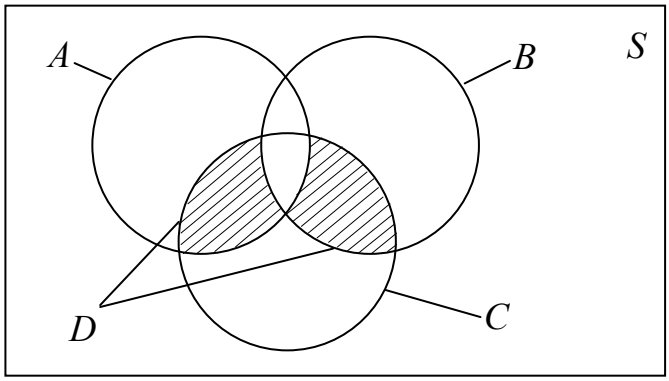


Рис. 1.11. Графічне подання умови задачі 5

6. Нехай A – множина студентів групи; B – множина студентів-відмінників тієї самої групи. Що означають такі твердження:

$$A - B = A,$$
$$A - B = \varnothing,$$
$$A \cap B = A?$$

7. На діаграмі Ейлера (рис.1.12) заштрихувати ділянку, відповідну такій множині: $D = (A \cap C) - B$.

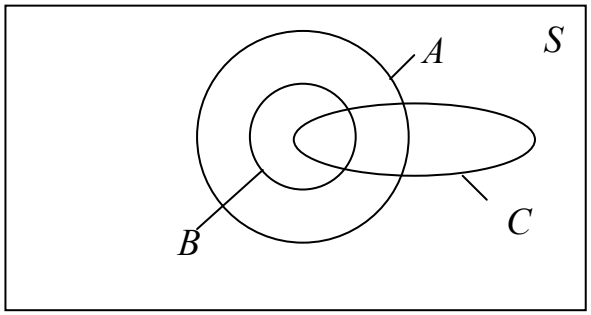


Рис.1.12. Графічне подання умови задачі 7

8. Дано множини: $A = Z$, $B = \{x | x \in R; -2 \le x < 2\}$, $C = \{x | x \in R; 0 < x \le 5\}$. Визначити множини:

$$D = A \cap B \cap C,$$
$$E = (A \cap C) - B,$$
$$F = A - (C \cup B),$$
$$G = (A \cap B) - C.$$

9. Знайти множини A та B, якщо:

а) $A \cap B = \{2,3\}; A \subset B; B - A = \{6\};$

б) $A \cup B = \{1,4,5,7,9\}; B - A = \varnothing; A - B = \{4,7,9\}.$

10. Дано множини: $A = \{2, 3, 5, 6, 8\}$, $B = \{1, 3, 5, 7\}$, $C = \{4, 6\}$. Виразити через них такі множини: $D = \{2, 8\}$, $E = \{1, 6\}$.

РОЗДІЛ 2
ЕЛЕМЕНТИ ТЕОРІЇ ГРАФІВ

Розглянуто основні поняття теорії графів і їх застосування до розв'язування оптимізаційних задач.

Багато структур, цікавих із позицій математики, інформатики та їх практичного застосування можуть бути описані за допомогою *графів.* У загальному значенні *графом* називають непорожню множину вершин (вузлів), з'єднаних ребрами. Для різних галузей застосування графи можуть розрізнятися напрямами, обмеженнями на кількість зв'язків, додатковими відомостями про вершини та ребра. У строгому математичному визначенні граф являє собою пару множин: $G = (V, E)$, де V – підмножина будь-якої лічильної множини (вершини), а E – підмножина $V \times V$ (ребра або зв'язки).

Теорія графів являє собою розділ дискретної математики, який вивчає властивості графів.

Зазначимо, що ця теорія набула широкого застосування в різних галузях науки та виробництва. Наприклад, у геоінформаційних системах (ГІС). Тут існуючі або проектовані будинки, споруди, квартали і т. п. розглядаються як вершини графа (множина V), а дороги, інженерні мережі, лінії електропередачі та ін., що їх з'єднують, як ребра (множина E). Обчислення, проведені на такому графі, дають можливість, наприклад, знайти найкоротший об'їзний шлях або найближчий продуктовий магазин, спланувати оптимальний транспортний маршрут. Коли за вершини графа взяти роботи, які мають бути виконані, а ребра будуть відповідати порядку їх виконання, то використовуючи методи теорії графів, можна, зокрема, визначити оптимальний порядок проведення робіт.

У теорії графів існує велика кількість невирішених проблем і поки що не доведених гіпотез.

Розглянемо основні поняття теорії графів.

§ 2.1. Основні поняття теорії графів

Нехай дано множину X, яка включає елементи, називані *вершинами графа*, а також закон (правило) G, що встановлює відповідність між кожним елементом x множини X та її підмножинами. Тоді граф буде повністю визначено множиною X і відповідністю G.

П р и к л а д. Припустимо, що $X = \{x_1, x_2, x_3, x_4, x_5\}$, $G(x_1) = \{x_2, x_3\}$, $G(x_2) = \{x_1, x_3, x_5\}$, $G(x_4) = \{x_3, x_5\}$, $G(x_3) = \{x_1, x_2, x_4\}$, $G(x_5) = \{x_2, x_4\}$.

Цей запис означає, що граф містить п’ять вершин (множина *X*). Відповідність *G* задає множину ребер графа. Зокрема запис: $G(x_1)=\{x_2,x_3\}$, означає, що існують ребра, які з’єднують овершину x_1 з вершинами x_2 та x_3.

Часто граф зображують геометрично. Відповідне наведеному вище прикладу зображення графа показано на рис. 2.1.

Лінії, що з'єднують вершини називаються *ребрами*. Для їх позначення іноді використовують запис: $g(x_i,x_j)$, тоді говорять, що ребро *g інцидентне* вершинам x_i і x_j, а вершини x_i і x_j – *суміжні*. У наведеному на рис. 2.1 графі суміжними будуть, наприклад, вершини x_2 і x_1, x_2 і x_3, x_2 і x_5. У тойже час вершини x_2 та x_4 не будуть суміжними.

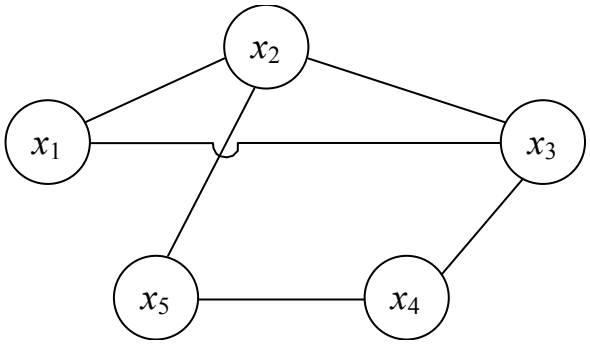
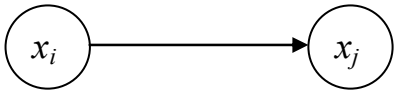


Рис. 2.1. Приклад геометричного зображення графа

Якщо порядок проходження вершин у записі $g(x_i,x_j)$ задано, то таке ребро називають *орієнтованим* або *дугою*, геометрично такий порядок позначають стрілкою, тобто:



У зв’язку з цим розрізняють *неорієнтовані*, тобто ті, що складаються із ребер, і *орієнтовані*, що складаються із дуг, *графи.*

Зробимо ще кілька визначень.

Повним називається граф, у якого будь-яка вершина з'єднується (ребрами або дугами) з усіма іншими вершинами. Так граф, зображений на рис. 2.1 не повний.

Петлею називається ребро (дуга) $g(x_i,x_j)$, у якої початкова й кінцева вершини збігаються.

Ланцюг – це послідовність ребер, які проходять через кілька вершин. Наприклад, послідовність ребер: $g(x_1,x_2)$, $g(x_2,x_3)$, $g(x_3,x_4)$ на рис. 2.1 являє собою ланцюг.

Ланцюг, у якого початкова й кінцеві вершини збігаються, будемо називати *циклом*. Ребра $g(x_1,x_2)$, $g(x_2,x_3)$, $g(x_3,x_1)$ на рис. 2.1 утворюють цикл.

Для орієнтованого графа мають місце аналогічні поняття «*шлях*», він являє собою послідовність дуг, які проходять через кілька вершин, і «*контур*» – це шлях, у якого початкова й кінцеві вершини збігаються.

Один і той самий граф можна зобразити по-різному. Так, на рис. 2.2 подано три варіанти зображення одного графа. Ця властивість графів називається *ізоморфізмом*.

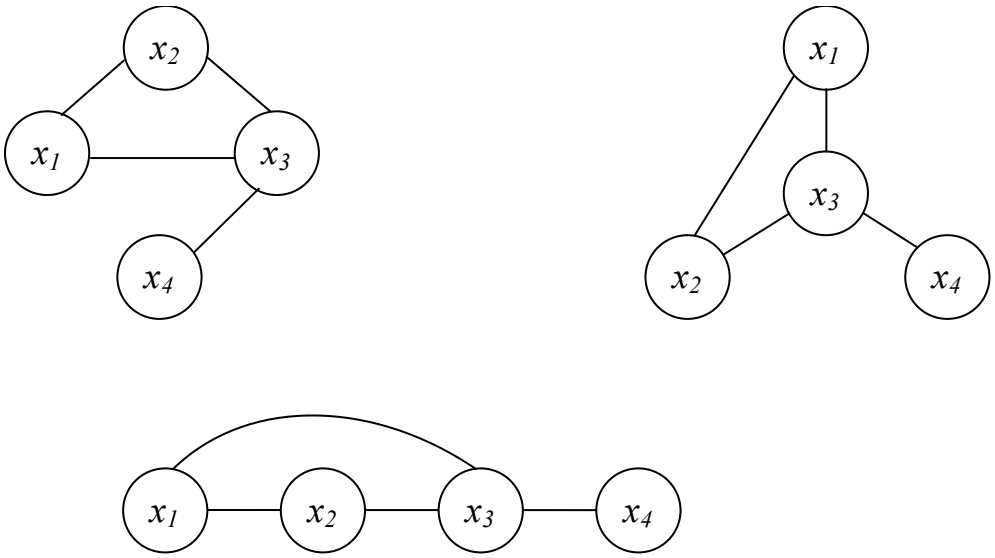


Рис. 2.2. Приклади ізоморфних графів

Взагалі, кожен граф можна задати трьома способами: графічно, аналітично й за допомогою матриць. Розглянемо їх докладніше.

1. *Графічний спосіб*. Він полягає в тому, що вершини графа зображуються точками або колами, його ребра (або дуги) – лініями (стрілками). Приклад такого задання графа наведено на рис. 2.3. Тут символами x_i позначено вершини графа, g_j – ребра або дуги. На рис. 2.3, *а* зображено неорієнтований, а на рис. 2.2, *б* – орієнтований графи.

а *б*

Рис. 2.3. Графічний способ задання графа

2. *Аналітичний спосіб*. Включає опис множини X та відповідності G. Два графи із попереднього прикладу можуть бути задані аналітично таким чином:

а) неорієнтований граф: $X=\{x_1,x_2,x_3,x_4\}$, $G(x_1)=\{x_2,x_3\}$, $G(x_2)=\{x_1,x_3\}$, $G(x_3)=\{x_1,x_2,x_4\}$, $G(x_4)=\{x_3\}$;

б) орієнтований граф: $X=\{x_1x_2x_3x_4\}$, $G(x_1)=\{x_2x_4\}$, $G(x_2)=\{x_4\}$, $G(x_3)=\{x_2\}$, $G(x_4)=\{x_3\}$.

3. *Матричний спосіб*. Граф може бути заданий також у зручному для машинної обробки вигляді за допомогою матриць суміжності або інциденцій.

Матрицею суміжності називається квадратна матриця A, розмір якої дорівнює числу вершин графа, а кожний елемент a_{ij} визначається таким чином: він дорівнює одиниці, якщо з вершини x_i можна безпосередньо потрапити у вершину x_j, і нулю у протилежному випадку. У нашому прикладі матриці суміжності мають такий вигляд:

$$A_{нор}=\begin{pmatrix}0&1&1&0\\1&0&1&0\\1&1&0&1\\0&0&1&0\end{pmatrix}$$ – для неорієнтованого графа;

$$A_{ор}=\begin{pmatrix}0&1&0&1\\0&0&0&1\\0&1&0&0\\0&0&1&0\end{pmatrix}$$ – для орієнтованого.

Зауважимо, що матриця суміжності неорієнтованого графа завжди симетрична.

Позначимо ребра (дуги) графа через $g_1, g_2, \dots g_m$, а вершини через $x_1, x_2, \dots x_n$.

Матрицею інциденцій ребер графа називається матриця: $R=\|r_{ij}\|_{n\times m}$, кожний елемент якої визначено в такий спосіб:

$$r_{ij}=\begin{cases}1, \textit{ коли вершина } x_i \textit{ інцидентна ребру } g_j,\\ 0, \textit{ якщо ні.}\end{cases}$$

Матрицею інциденцій дуг графа називається матриця: $S=\|s_{ij}\|_{n\times m}$, кожний елемент якої визначається таким чином:

$$s_{ij}=\begin{cases} 1, \textit{ коли ребро } g_j \textit{ виходить з вершини } x_i, \\ -1, \textit{ якщо ребро } g_j \textit{ заходить у вершину } x_i, \\ 0, \textit{ коли ребро } g_j \textit{ не інцидентне вершині } x_i. \end{cases}$$

У нашому прикладі

$$R=\begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \qquad S=\begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 & -1 \end{pmatrix}.$$

При розв'язуванні практичних задач ребрам або дугам графа ставиться у відповідність певне число $(x_i x_j)$, яке в кожному конкретному випадку відображає ту чи іншу характеристику, наприклад: відстань, витрати, пропускну здатність, часовий інтервал між подіями і т. д.

Граф, кожній дузі якого відповідає певне число, називається *зваженим*.

Розглянемо типові оптимізаційні задачі на графах і методи їх розв'язування.

§ 2.2. Задача про найкоротший шлях у графі

Наведемо приклади практичних задач, розв’язок яких може бути зведено до пошуку найкоротшого шляху в графі.

Приклад 1. Маємо мережу автомобільних доріг, яка з'єднує міста Дніпропетровської області. Деякі з них односторонні. Знайти найкоротші шляхи від Дніпропетровська до кожного міста області (якщо рухатися можна тільки по дорогах).

Приклад 2. Наявна деяка кількість авіарейсів між містами світу, вартість кожного з них відома. До того ж вартість перельоту з міста A в місто B не обов’язково дорівнює вартості перельоту у зворотному напрямку. Визначити маршрут між двома заданими містами (можливо з пересадками), який має мінімальну вартість.

Сформулюємо цю задачу математично.

Кожній дузі (x,y) вихідного графа G поставимо у відповідність число $a(x,y)$. Якщо в графі G відсутня деяка дуга (x,y), то припустимо, що $a(x,y)=\infty$. Будемо називати число $a(x,y)$ *довжиною дуги* (x,y), хоча $a(x,y)$ можна інтерпретувати і як відповідні витрати, або ваговий коефіцієнт.

Довжиною шляху будемо вважати суму значень довжин окремих дуг, які створюють цей шлях.

Для будь-яких двох вершин *s* і *t* графа *G* можуть існувати кілька шляхів, які з'єднують вершину *s* із вершиною *t*. Необхідно визначити шлях мінімальної довжини, який з'єднує вершину *s* і вершину *t*. Його називають *найкоротшим* шляхом між вершинами *s* і *t*.

Нижче буде розглянуто алгоритм, який дозволяє визначити такий шлях.

Алгоритм пошуку найкоротшого шляху (алгоритм Дейкстри)

Ідея алгоритму. Кожній вершині з множини *X* поставимо у відповідність число *d*(*x*) – найменшу відому відстань від цієї вершини до *s*. Алгоритм включає кілька кроків, зокрема на кожному кроці опрацьовується одна вершина з метою зменшення чисел *d*(*x*). Робота алгоритму закінчується, коли всі вершини опрацьовано.

Опишемо цей алгоритм.

Крок 1. *Ініціалізація*.

Задамо, що $d(s)=0$ і $d(x)=\infty$ для всіх вершин *x*, відмінних від *s* (це відображає той факт, що відстані до цих вершин на даний момент не відомі). Жодну вершину ще не опрацьовано (не забарвлено). Через *y* позначимо останню із забарвлених вершин. Позначимо (пофарбуємо) вершину *s* і приймемо, що $y = s$.

Крок 2. *Основний крок алгоритму*.

Для кожної незабарвленої вершини *x* перерахуємо величину відстані $d(x)$ в такий спосіб:

$$d(x)=\min\{d(x);d(y)+a(y,x)\}. \qquad (2.1)$$

Якщо $d(x)=\infty$ для всіх незабарвлених вершин *x*, то належить закінчити процедуру алгоритму, оскільки у вихідному графі відсутні шляхи з вершини *s* у незабарвлені вершини.

Коли ні, то слід пофарбувати ту з вершин *x*, для якої величина $d(x)$ найменша. Крім того, потрібно пофарбувати дугу, що веде до обраної на даному кроці вершини *x* [саме для цієї дуги досягався мінімум виразу (2.1)]. Прийняти, що $y = x$.

Крок 3. Якщо $y = t$, то процедуру закінчити, оскільки найкоротший шлях від вершини *s* до вершини *t* знайдений (це єдиний шлях від *s* до *t*, який складається із забарвлених дуг). А якщо ні, то перейти до *кроку* 2.

Алгоритм описано.

Забарвлені дуги утворюють у вихідному графі орієнтоване дерево з коренем у вершині *s*. Його називають *орієнтованим деревом найкоротших шляхів.* Єдиний шлях від вершини *s* до будь-якої вершини *x*, що належить

дереву найкоротших шляхів, буде найкоротшим шляхом між зазначеними вершинами.

Оскільки на всіх етапах алгоритму Дейкстри забарвлені шляхи утворюють у вихідному графі орієнтоване дерево, то алгоритм можна розглядати як процедуру нарощування орієнтованого дерева з коренем у вершині *s*. Коли в процедурі нарощування буде досягнуто вершини *t*, процедуру припиняють.

Для визначення найкоротших шляхів від вершини *s* до всіх вершин вихідного графа процедуру нарощування дерева слід продовжити доти, поки всі вершини графа не будуть включені в орієнтоване дерево найкоротших шляхів. При цьому для вихідного графа буде отримано *покривне дерево* (звісно, якщо в даному графі існує хоча б одне таке дерево).

Отже, для того, щоб описаний вище алгоритм дозволив одержати дерево найкоротших шляхів від вершини *s* до всіх інших вершин, його третій крок повинен бути скорегований у такий спосіб:

Крок 3а. Якщо всі вершини виявляються забарвленими, то процедуру закінчують (існує єдиний найкоротший шлях від вершини *s* до будь-якої вершини *x*, і він складається тільки із забарвлених дуг) А якщо ні, то повертаються до *кроку* 2.

Зауважимо, що алгоритм Дейкстри застосовується при роботі з орієнтованими графами, які не мають петель, а довжини дуг не набувають від'ємних значень. Але, він може бути узагальнений на випадок, коли деякі з дуг мають від'ємні значення довжини. Необхідна модифікація описаного вище алгоритму (Форда) передбачає такі дії:

– На *другому кроці* алгоритму перерахування величини відстані $d(x)$ за допомогою співвідношення (2.1) проводиться для всіх вершин, а не тільки для незабарвлених. Отже, числа $d(x)$ можуть зменшуватися як для незабарвлених, так і для пофарбованих вершин.

– Якщо для деякої пофарбованої вершини *x* відбувається зменшення величини $d(x)$, то із цієї вершини та з інцидентної їй пофарбованої дуги фарбування знімається.

– Процедура алгоритму закінчується тільки тоді, коли всі вершини пофарбовані й коли після виконання кроку 2 жодне із чисел $d(x)$ не змінюється.

Приклад 1. За допомогою алгоритму Дейкстри визначити найкоротший шлях між вершинами *s* і *t* у графі, зображеному на рис. 2.4.

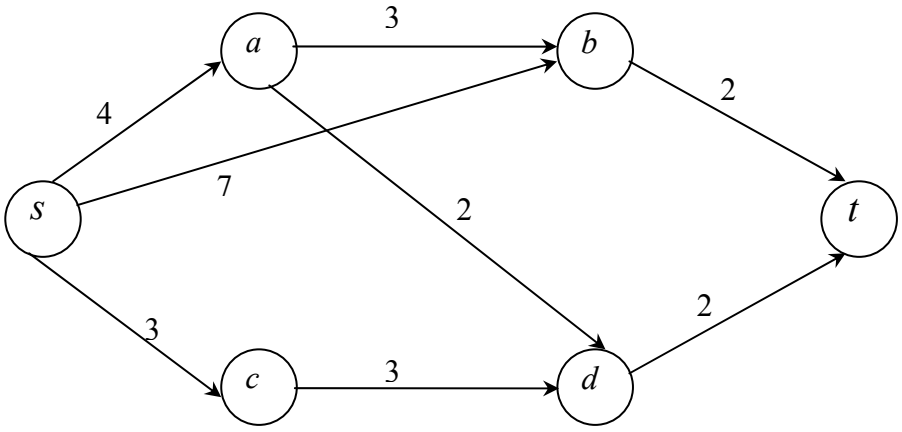


Рис. 2.4. Граф до прикладу 1

Розв'язування

Крок 1. Зафарбуємо вершину *s*. Приймемо, що $d(s)=0$ і $d(a)=d(b)=d(c)=d(d)=d(t)=\infty;\ (y=s)$. На графі запишемо значення величин $d(x)$ поряд із вершиною. Тоді він набуває поданого на рис. 2.5 вигляду.

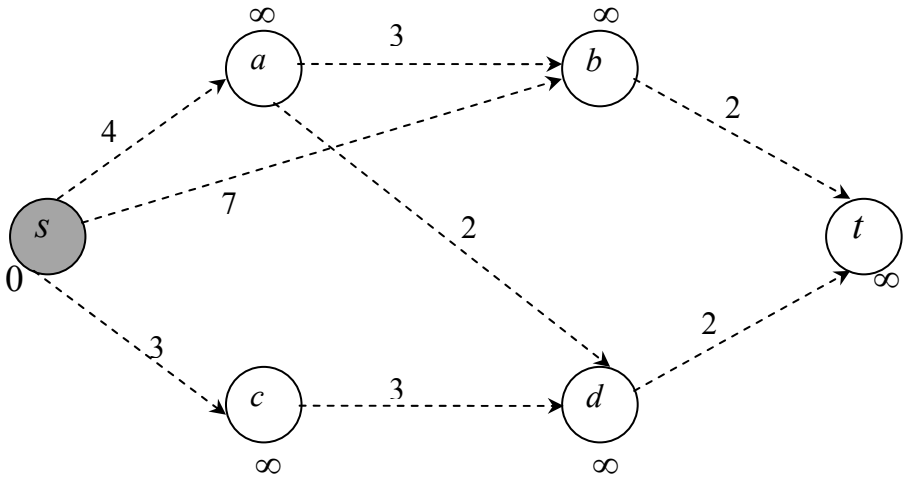


Рис. 2.5. Перший крок алгоритму пошуку найкоротшого шляху

Крок 2. Для незабарвлених вершин зробимо перерахування відстаней $d(x)$, а саме:

$$d(a)=\min\{d(a),d(s)+a(s,a)\}=\min\{\infty,0+4\}=4;$$

$$d(b)=\min\{d(b),d(s)+a(s,b)\}=\min\{\infty,0+7\}=7;$$

$$d(c)=\min\{d(c),d(s)+a(s,c)\}=\min\{\infty,0+3\}=3;$$

$$d(d)=\min\{d(d),d(s)+a(s,d)\}=\min\{\infty,0+\infty\}=\infty;$$

$$d(t)=\min\{d(t),d(s)+a(s,t)\}=\min\{\infty,0+\infty\}=\infty.$$

Оскільки $\min\{d(a),d(b)d(c),d(d),d(t)\}=\min\{4,7,3,\infty,\infty\}=3=d(c)$, то зафарбовуємо вершину c і дугу (s,c). Поточне дерево найкоротших шляхів показано на графі (рис. 2.6) товстою лінією.

Крок 3. Через те, що вершина t залишилася незабарвленою, повертаємось до кроку 2, прийнявши, що $y=c$.

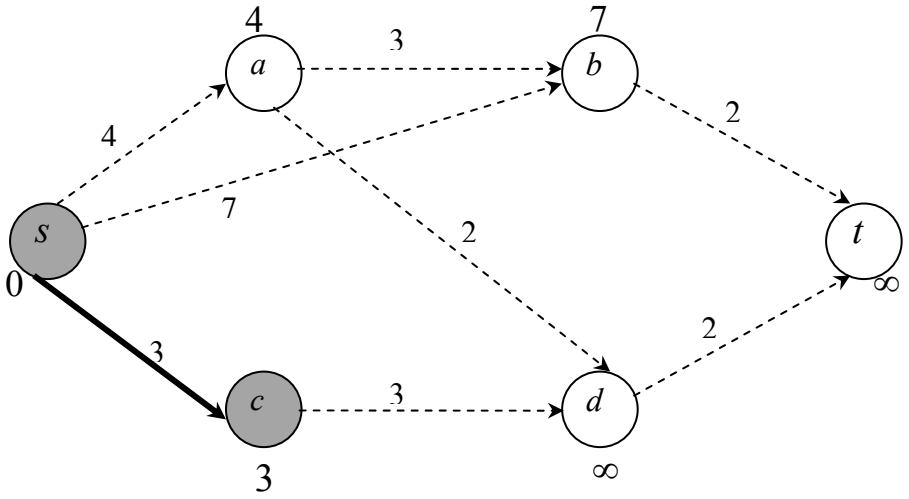


Рис. 2.6. Побудова дерева найкоротших шляхів (крок 2, $y=s$)

Крок 2 $(y=c)$.

Перераховуємо величини $d(x)$ для незабарвлених вершин x, а саме:

$$d(a)=\min\{d(a),d(c)+a(c,a)\}=\min\{4,3+\infty\}=4;$$

$$d(b)=\min\{d(b),d(c)+a(c,b)\}=\min\{7,3+\infty\}=7;$$

$$d(d)=\min\{d(d),d(c)+a(c,d)\}=\min\{\infty,3+3\}=6;$$

$$d(t)=\min\{d(t),d(c)+a(c,t)\}=\min\{\infty,3+\infty\}=\infty.$$

Оскільки $\min\{d(a),d(b),d(d),d(t)\}=\min\{4,7,6,\infty\}=4=d(a)$, то зафарбовуємо вершину a й дугу (s,a). Поточне дерево найкоротших шляхів набуває вигляду, зображеного на рис. 2.7.

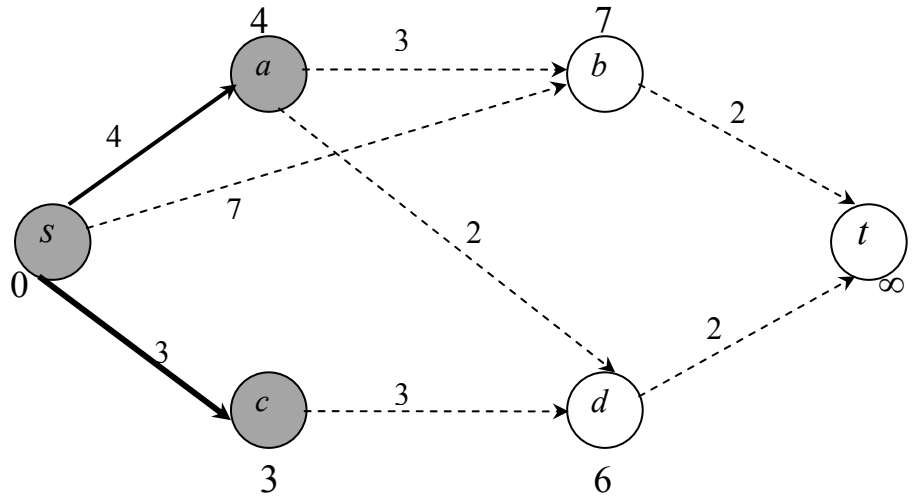


Рис. 2.7. Побудова дерева найкоротших шляхів (крок 2, $y=c$)

Крок 3**.** Вершина t залишилася незабарвленою, тому повертаємось до кроку 2, прийнявши, що $y=a$.

Крок 2 $(y=a)$.

Для незабарвлених вершин робимо перерахування чисел $d(x)$, а саме:

$$d(b)=\min\{d(b),d(a)+a(a,b)\}=\min\{7,4+3\}=7;$$

$$d(d)=\min\{d(d),d(a)+a(a,d)\}=\min\{6,4+2\}=6;$$

$$d(t)=\min\{d(t),d(a)+a(a,t)\}=\min\{\infty,4+\infty\}=\infty.$$

Враховуючи, що $\min\{d(b),d(d),d(t)\}=\min\{7,6,\infty\}=6=d(d)$, зафарбовуємо вершину d й дугу (a,d). Зауважимо, що можна було зафарбувати дугу (c,d).

Поточне дерево найкоротших шляхів набуває вигляду, показаного на рис. 2.8.

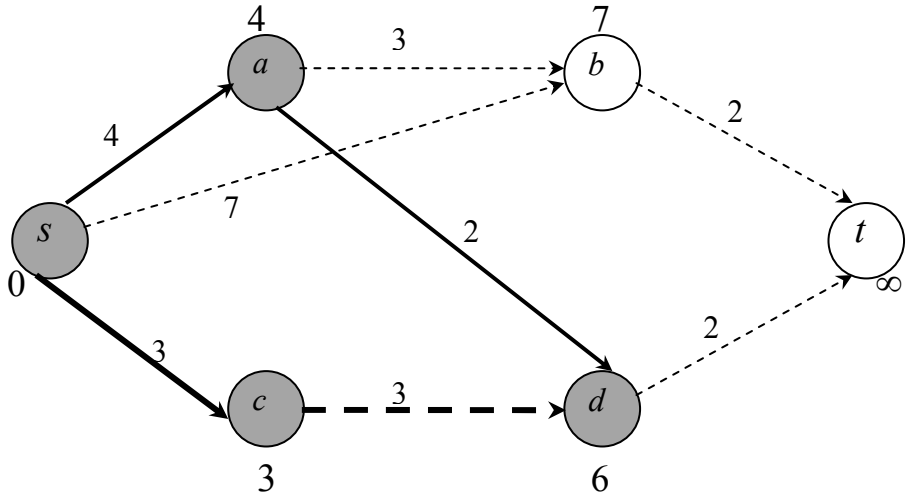


Рис. 2.8. Поточне дерево найкоротших шляхів (крок 2, $y=a$)

Крок 3. Оскільки вершина t залишилася незабарвленою, то знов повертаємось до кроку 2, вважаючи, що $y = d$.

Крок 2 $(y = d)$.

Для незабарвлених вершин робимо перерахування відстаней $d(x)$:

$$d(b) = \min\{d(b), d(d) + a(d,b)\} = \min\{7, 6 + \infty\} = 7;$$

$$d(t) = \min\{d(t), d(d) + a(d,t)\} = \min\{\infty, 6 + 2\} = 8.$$

Оскільки $\min\{d(b), d(t)\} = \min\{7,8\} = 7 = d(b)$, то зафарбовуємо вершину b й дугу (s,b). Можна було б також зафарбувати дугу (a,b).

Поточне дерево найкоротших шляхів набуває вигляду, поданого на рис. 2.9.

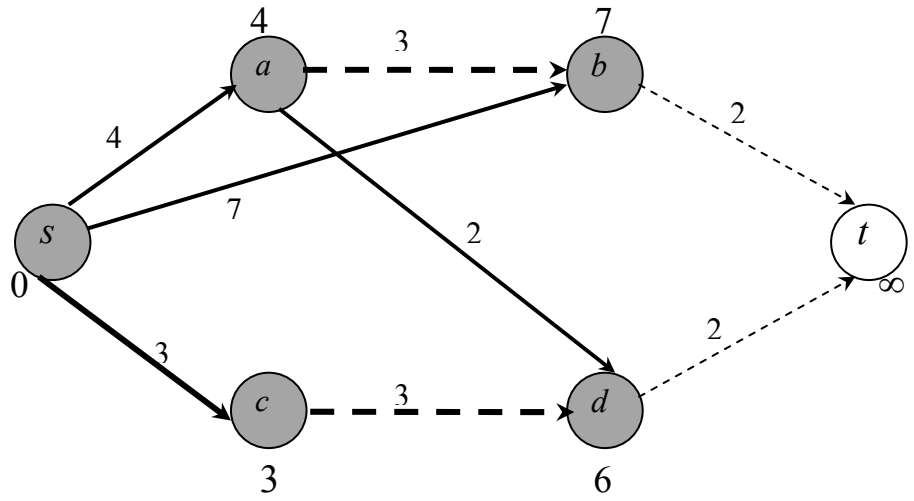


Рис. 2.9. Поточне дерево найкоротших шляхів (крок 2, $y = d$)

Крок 3. Вершина t залишилася незабарвленою, тому знову повертаємось до кроку 2, приймаючи, що $y = b$.

Крок 2. $(y = b)$.

Для незабарвлених вершин робимо перерахування чисел $d(x)$, а саме:

$$d(t) = \min\{d(t), d(b) + a(b,t)\} = \min\{8, 7 + 2\} = 8.$$

Отже, вершину t, нарешті, можна пофарбувати. Разом з нею фарбуємо й дугу (d,t), яка визначає величину $d(t)$.

Остаточно побудоване дерево найкоротших шляхів має вигляд, показаний на рис. 2.10.

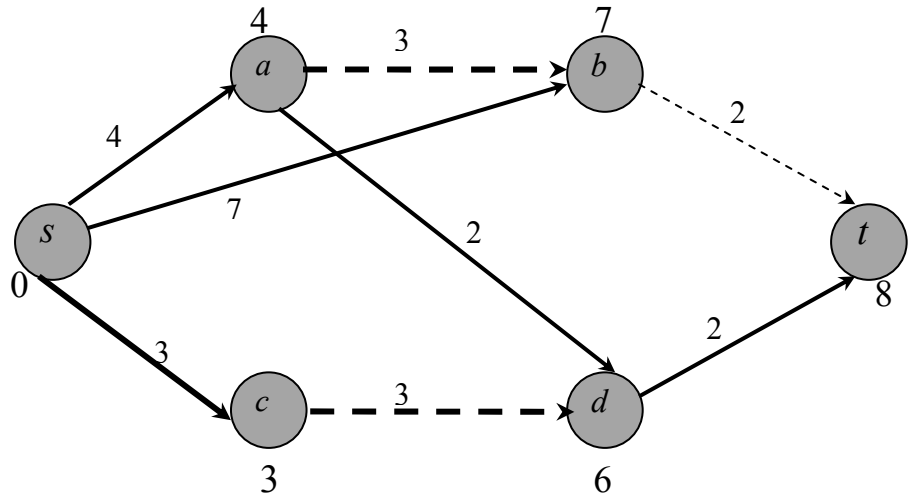


Рис. 2.10. Дерево найкоротших шляхів у графі з прикладу 1

Очевидно, що у вихідному графі є два найкоротші шляхи однакової довжини: перший шлях складається з дуг $(s,a),(a,d),(d,t)$; другий шлях включає дуги $(s,c),(c,d),(d,t)$.

Зауважимо, що найкоротший шлях буде єдиним тільки тоді, коли під час процедури алгоритму жодного разу не виникає неоднозначності у виборі забарвленої дуги.

Приклад 2. Застосуємо алгоритм Форда до графа, поданого на рис. 2.11.

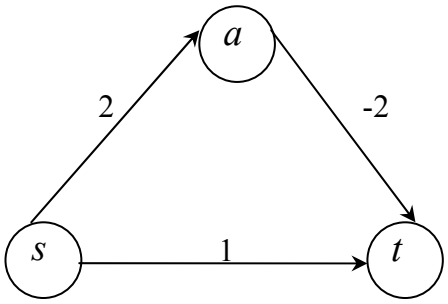


Рис. 2.11. Граф до прикладу 2

Розв'язування

Крок 1. Зафарбуємо вершину *s*. Задамо, що $d(s)=0$ і $d(a)=d(t)=\infty$, $y=s$. Позначимо величини $d(x)$ поряд з вершинами графа (рис. 2.12).

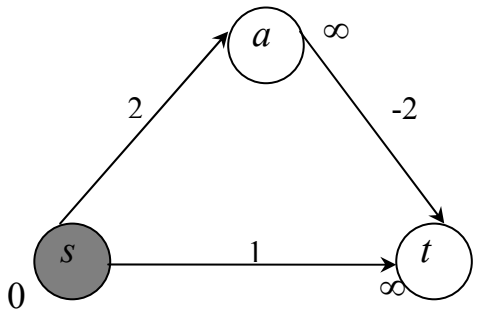


Рис. 2.12. Крок 1 алгоритму Форда в прикладі 2

Крок 2 $(y = s)$.

Для незабарвлених вершин *x* робимо перерахування чисел $d(x)$, а саме:

$$d(a) = \min\{d(a), d(s) + a(s,a)\} = \min\{\infty, 0+2\} = 2;$$

$$d(t) = \min\{d(t), d(s) + a(s,t)\} = \min\{\infty, 0+1\} = 1.$$

Беручи до уваги, що $\min\{d(a), d(t)\} = 1 = d(t)$, зафарбовуємо вершину *t* й дугу (s,t). Поточне дерево найкоротших шляхів набуває вигляду, зображеного на рис. 2.13.

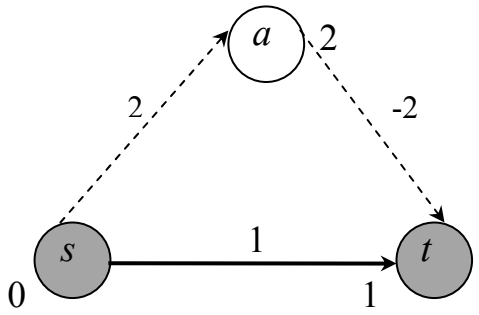


Рис. 2.13. Крок 2 алгоритму Форда у прикладі 2 $(y = s)$

Крок 3. Оскільки забарвленими виявилися не всі вершини, повертаємось до кроку 2, вважаючи, що $y = t$.

Крок 2 $(y = t)$

Враховуючи те, що з вершини *t* не виходить жодна дуга, усі числа $d(x)$ залишаються незмінними, тому фарбуємо вершину *a* і разом з нею дугу (s,a). Поточне дерево найкоротших шляхів позначено на графі (див. рис. 2.14) товстою лінією.

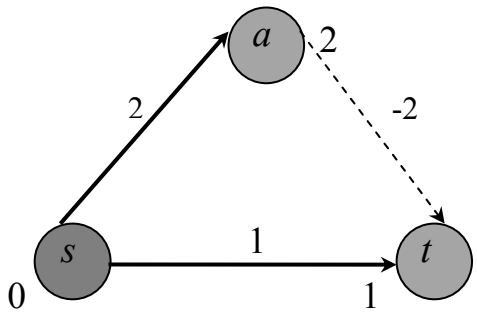


Рис. 2.14. Крок 2 алгоритму Форда в прикладі 2 $(y=t)$

Крок 3. Здійснюємо повернення до кроку 2, аби спробувати зменшити значення чисел $d(x)$.

Крок 2 $(y=a)$.

Обчислюємо, що

$$d(t)=\min\{d(t),d(a)+a(a,t)\}=\min\{1,2-2\}=0,$$

$$d(s)=\min\{d(s),d(a)+a(a,s)\}=\min\{0,2+\infty\}=0.$$

Оскільки величина $d(t)$ зменшується від 1 до 0, то з вершини *t* і дуги (s,t) фарбування знімається.

Тепер поточне дерево найкоротших шляхів має вигляд, показаний на рис. 2.15.

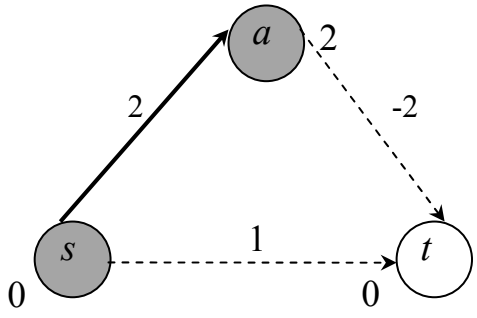


Рис. 2.15. Крок 2 алгоритму Форда в прикладі 2 $(y=a)$

У вихідному графі залишається незабарвленою тільки вершина *t* . Зафарбовуємо її разом з дугою (a,t), оскільки на даному кроці $y=a$.

Дерево найкоротших шляхів тепер набуває вигляду, зображеного на рис. 2.16.

Крок 3. Здійснюємо чергове повернення до кроку 2.

Крок 2 $(y=t)$

Враховуючи те, що з вершини *t* не виходить жодна дуга, значення чисел $d(x)$ не змінюються. Незабарвлених вершин у графі немає.

Крок 3. З огляду на те, що всі вершини в графі виявляються забарвленими й на попередньому кроці алгоритму жодну з величин $d(x)$ зменшити не вдалося, процедура алгоритму завершується, найкоротший шлях з вершини *s* у вершину *t* складається з дуг $(s,a),(a,t)$ і має таку довжину: 2 – 2 = 0 (див. рис. 2.16).

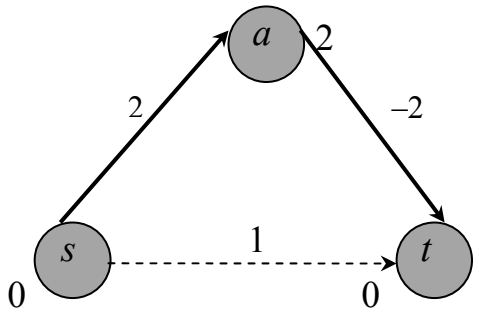


Рис. 2.16. Найкоротший шлях у графі з прикладу 2

Зауважимо, що алгоритм Форда не забезпечує розв'язування зазначеної задачі за умови наявності у вихідному графі контуру, довжина якого становить від'ємну величину. Дійсно, у цьому випадку можна, необмежене число раз «повторюючи» цей контур, одержати шлях як завгодно малої довжини.

У ситуації, коли немає даних про відсутність або наявність у розглянутому графі контурів від'ємної довжини, можна застосовувати алгоритм Форда, але в процесі його застосування необхідно враховувати, скільки разів забарвлюються окремі вершини. Коли число фактів забарвлення однієї з вершин досягає величини *N*, де *N* – число вершин графа, процедуру алгоритму можна зупинити, бо вихідний граф містить контур, довжина якого має від'ємне значення. Коли ж цього не відбувається, то процедура алгоритму Форда завершується протягом скінченної кількості кроків, а вихідну задачу буде розв'язано правильно.

§ 2.3. Задача побудови мінімального кістякового дерева

Одним з поширених практичних питань, які в математичному формулюванні являють собою задачу про знаходження мінімального кістякового (покривного) дерева, буде таке: припустимо, існує *n* міст, які необхідно з'єднати дорогами, таким чином, щоб можна було дістатися з кожного міста в будь-яке інше (прямим сполученням або через інші міста). Дозволяється будувати дороги між заданими парами міст, причому відома вартість будівництва кожної такої дороги. Необхідно вирішити, які саме дороги потрібно прокладати, щоб мінімізувати загальну вартість будівництва.

З огляду на теорію графів, ця задача може бути сформульована в такий спосіб: знайти мінімальне кістякове дерево в графі, вершинам якого поставлено у відповідість міста, а ребра відповідають парам міст, між якими можна прокласти пряму дорогу. При цьому вага ребра дорівнює вартості будівництва відповідної дороги.

Сформульована задача розв’зується досить просто. Виконання алгоритму починається з вибору довільного вузла мережі та найкоротшої дуги із множини дуг, що з'єднують цей вузол з іншими вузлами. Після цього з'єднуємо два вузли обраною дугою і вибираємо найближчий до них третій вузол. Далі додаємо цей вузол і відповідну дугу до шуканої мережі. Продовжуємо діяти таким чином доти, поки всі вузли не будуть з'єднані між собою. Алгоритм, що базується на «поглинанні» найкоротших дуг, описуємо нижче.

Алгоритм побудови мінімального кістякового дерева

Крок 1. Використовуючи вузли (вершини) вихідної мережі, визначаємо такі дві множини:

s – множину з'єднаних вузлів;

$\overline{s}$ – множину несполучених вузлів.

Спочатку всі вузли будуть належати множині $\overline{s}$.

Крок 2. Із множини $\overline{s}$ вибираємо довільний вузол і з'єднуємо його з найближчим сусіднім вузлом (після виконання даного кроку множина s буде містити два вузли).

Крок 3. Серед усіх дуг, що з'єднують вузли множини s із вузлами множини $\overline{s}$, вибираємо найкоротшу дугу. Кінцевий вузол цієї дуги, який перебуває в множині $\overline{s}$, позначимо через δ. Вилучаємо вузол δ із множини $\overline{s}$ і поміщаємо його в множину s.

Крок 4. Виконуємо крок 3, поки всі вузли не будуть належати множині s.

Алгоритм описано.

Приклад 3. Знайти мінімальне дерево в мережі, зображеній на рис. 2.17.

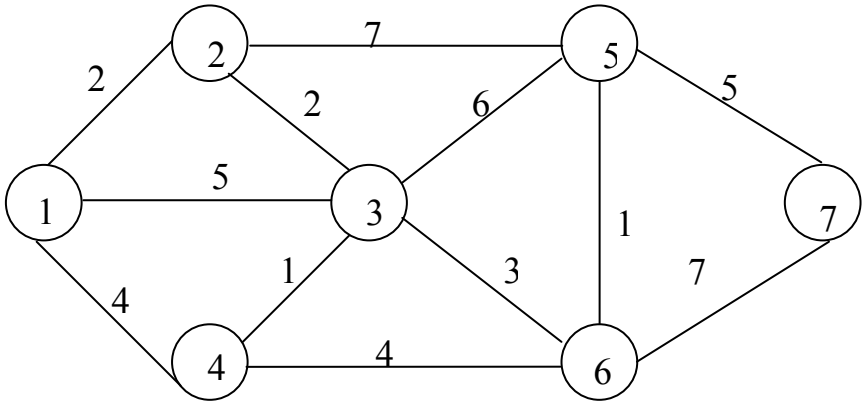


Рис. 2.17. Мережа до прикладу 3

Розв'язування

Крок 1. Визначаємо множини s, $\overline{s}$, а саме:

$$\overline{s} = \{1, 2, 3, 4, 5, 6, 7\}, \; s = \varnothing.$$

Крок 2. Вибираємо для розгляду вузол 6, тоді $s = \{6, 5\}$, $\overline{s} = \{1, 2, 3, 4, 7\}$.

Побудована на цьому кроці мережа на рис. 2.18. зображена товстою лінією. Вартість побудованої мережі $p = 1$.

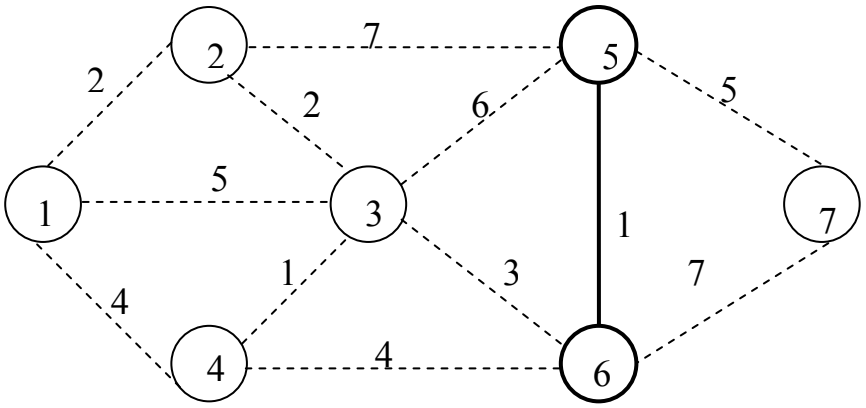


Рис. 2.18. Перший етап побудови мінімального кістякового дерева

Крок 3.

1. Вибираємо вузол 3. Множини матимуть такий вигляд: $s = \{6,5,3\}$, $\overline{s} = \{1,2,4,7\}$. Вартість побудованої мережі $p = 1 + 3 = 4$ одиниці, її вигляд показано на рис. 2.19.

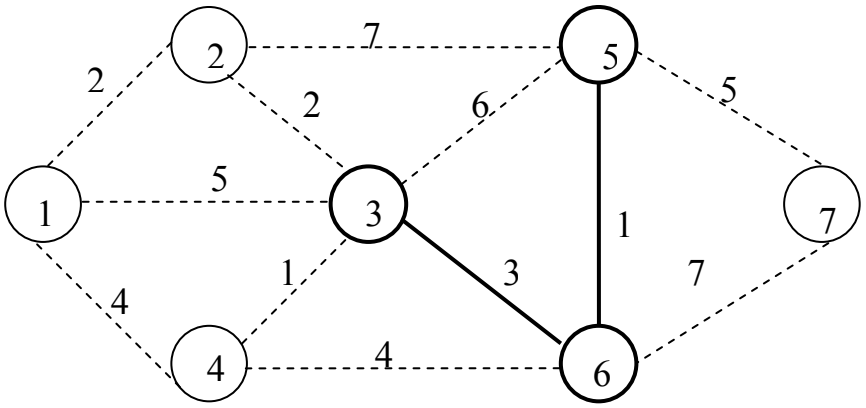


Рис. 2.19. Побудова мінімального кістякового дерева (крок 3, п. 1)

2. Вибираємо вузол 4, тоді $s = \{6,5,3,4\}$, $\overline{s} = \{1,2,7\}$. Вартість побудованої мережі $p = 1 + 3 + 1 = 5$ одиниць (див. рис. 2.20).

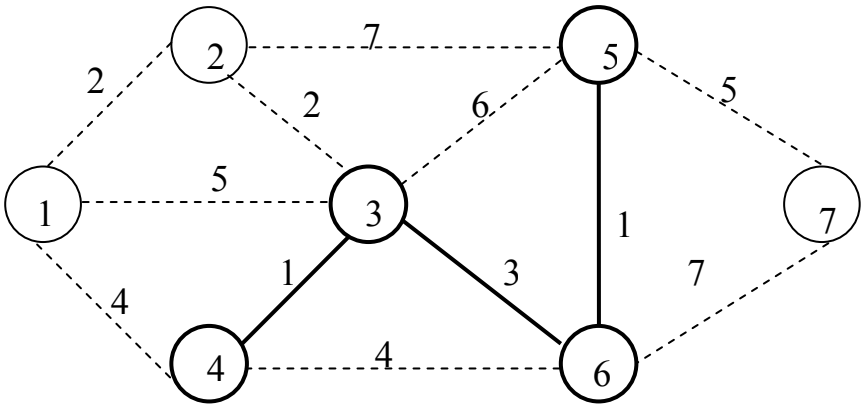


Рис. 2.20. Побудова мінімального кістякового дерева (крок 3, п. 2)

3. Вибираємо вузол 2. У цьому випадку множини: $s=\{6,5,3,4,2\}$, $\overline{s}=\{1,7\}$. Вартість побудованої мережі $p = 1 + 3 + 1 + 2 = 7$ одиниць (див. рис. 2.21).

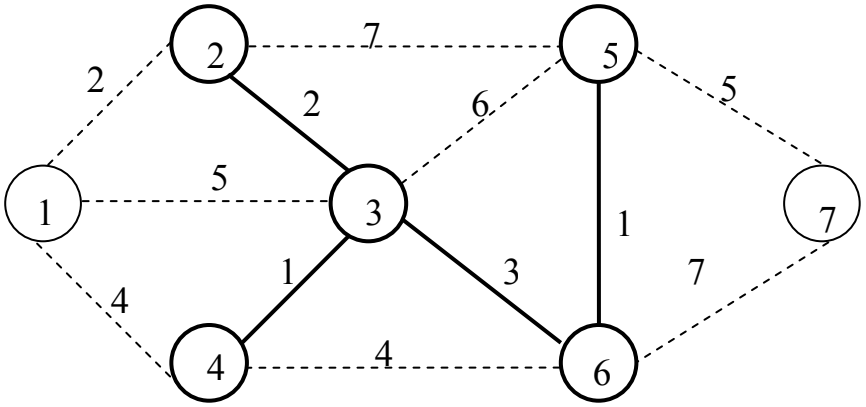


Рис. 2.21. Побудова мінімального кістякового дерева (крок 3, п. 3)

4. Вибираємо вузол 1, тоді $s=\{6,5,3,4,2,1\}$, $\overline{s}=\{7\}$. Вартість побудованої мережі $p = 1 + 3 + 1 + 2 + 2 = 9$ одиниць (див. рис. 2.22).

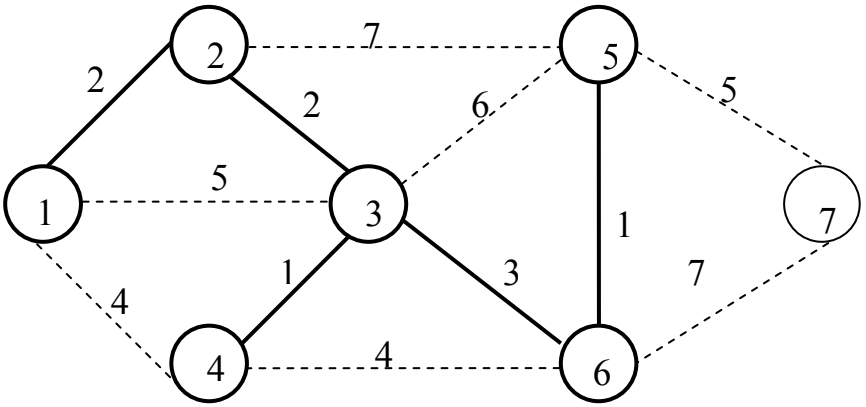


Рис. 2.22. Побудова мінімального кістякового дерева (крок 3, п. 4)

5. Вибираємо вузол 7, тоді $s=\{6,5,3,4,2,1,7\}$, $\overline{s}=\{\varnothing\}$. Вартість побудованої мережі p = 1 + 3 +1 + 2 + 2 +5 = 14 одиниць. Його вигляд подано на рис. 2.23.

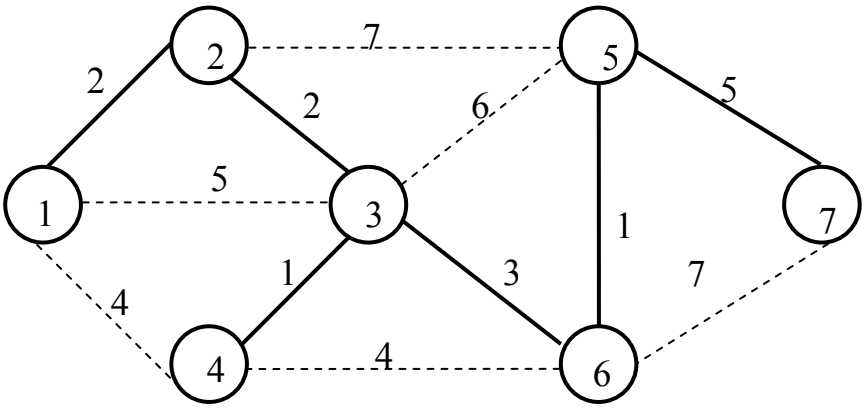


Рис. 2.23. Мінімальне кістякове дерево мережі з прикладу 3

Оскільки множина $\overline{s}=\varnothing$, то виконання алгоритму завершено. Вартість мінімального кістякового дерева дорівнює 14 одиниць.

§ 2.4. Задача про максимальний потік у графі

Перш, ніж розглядати постановку задачі й алгоритми її розв'язку, уведемо деякі визначення.

Вершина, з якої починається переміщення об'єктів, називається *джерелом* і звичайно позначається через s.

Вершина, у якій закінчується переміщення об'єктів, називається *стоком* і звичайно позначається через t.

Об'єкти, які переміщаються або «течуть» із джерела в стік, називаються *одиницями потоку*, або просто *одиницями*.

Коли кількість одиниць потоку, які можуть проходити по дузі (x,y), обмежена, то говорять, що дуга (x,y) має обмежену *пропускну здатність*.

Задача про максимальний потік полягає у відшукуванні способу пересилання максимальної кількості одиниць із джерела в стік без перевищення пропускної здатності кожної дуги вихідного графа.

Максимальну величину пропускної здатності будемо позначати через $c(x,y)$.

Кількість одиниць, що проходять по дузі (x,y), прийнято називати *потоком* у даній дузі. Позначимо його через $f(x,y)$. Очевидно, що $0\le f(x,y)\le c(x,y)$.

Дуги графа можна віднести до трьох різних категорій, а саме:

– ті, у яких потік не може ні збільшуватися ні зменшуватися (множина таких дуг позначається через N);

– ті, у яких потік може збільшуватися (множина таких дуг позначається через I);

– ті, у яких потік може зменшуватися (множина таких дуг позначається через R).

Дуги із множини I називаються *збільшувальними*, а дуги із множини R – *зменшувальними*.

Уведемо такі позначення:

$i(x,y)$ – максимальна величина, на яку може бути збільшений потік у дузі(x,y), причому

$$i(x,y)=c(x,y)-f(x,y); \tag{2.1}$$

$r(x,y)$ – максимальна величина, на яку може бути зменшений потік у дузі (x,y), а саме:

$$r(x,y)=f(x,y). \tag{2.2}$$

Переслати додаткову кількість одиниць із джерела в стік можна кількома способами.

1. Існує шлях P від вершини *s* до вершини *t*, який цілком складається із збільшувальних дуг (див. рис. 2.24).

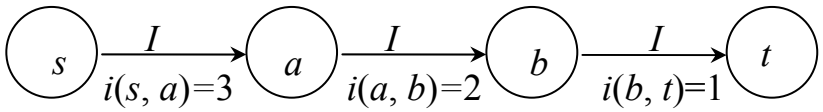


Рис. 2.24. Шлях, що цілком складається із збільшувальних дуг

Оскільки $i(x,y)$ являє собою максимально можливе збільшення потоку в дузі(x,y), то максимальна величина додаткового потоку, що переноситься з вершини *s* у вершину *t* по шляху *P*, буде визначатися величиною $\min\limits_{(x,y)\in P}\{i(x,y)\}$. Наприклад, для шляху, показаного на рис. 2.24. вона буде такою:

$$\min\{i(s,a),i(a,b),i(b,t)\}=\min\{3,2,1\}=1.$$

2. Існує шлях P від вершини *t* до вершини *s* , який цілком складається із зменшувальних дуг (рис. 2.25).

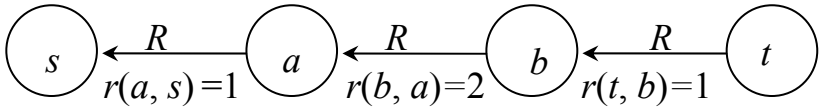


Рис. 2.25. Шлях, що цілком складається із зменшувальних дуг

Через те, що в кожній дузі (x,y) шляху P потік можна зменшити якнайбільше на величину $r(x,y)$, максимальне зменшення потоку вздовж шляху P визначається величиною $\min\limits_{(x,y)\in P}\{r(x,y)\}$.

Для даного прикладу (рис. 2.25) по шляху P можна переслати з вершини s у вершину t максимум одну одиницю потоку, оскільки

$$\min\{r(t,b),r(b,a),r(a,s)\}=\min\{1,2,1\}=1.$$

3. Цей спосіб поєднує в собі два перших. Тут проводиться пошук ланцюга, який з'єднує вершини s і t, і дуги якого задовольняють таким умовам:

– усі ті дуги ланцюга, що мають напрямок від вершини s до вершини t, тобто *прямі*, належать до множини I;

– усі ті дуги ланцюга, що мають напрямок від вершини t до вершини s, тобто *зворотні*, належать множині R.

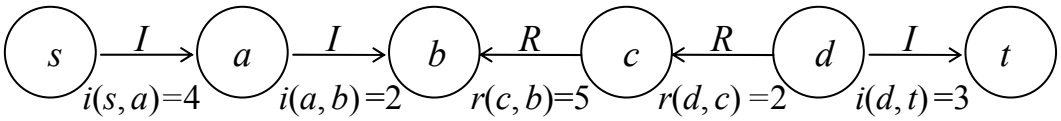


Рис. 2.26. Шлях, який включає збільшувальні й зменшувальні дуги

Переслати додатковий потік з вершини s у вершину t можна за рахунок збільшення потоку в прямих збільшувальних дугах, і зменшення в зворотних зменшувальних дугах.

Максимальна величина додаткового потоку, який можна передати вздовж такого ланцюга з вершини s у вершину t, визначається в такий спосіб:

$$\min\{\min\{i(s,a),i(a,b),i(d,t)\},\min\{r(c,b),r(d,c)\}\}=$$

$$=\min\{\min\{4,2,3\},\min\{2,5\}\}=2.$$

Ця величина називається *максимальним збільшенням потоку* ланцюга.

Усякий ланцюг (кожного із трьох типів, розглянутих вище), що веде від вершини s до вершини t і яким можуть бути додатково надіслані одиниці потоку, називається *збільшувальним ланцюгом.*

2.4.1. Алгоритм пошуку збільшувального ланцюга

Основна ідея алгоритму пошуку збільшувального ланцюга, полягає в побудові дерева, яке зростає від вершини s та складене із забарвлених дуг, по яких із неї можуть передаватися додаткові одиниці потоку. Опишемо цей алгоритм.

Крок 1. Визначаємо склад множин N, I, R. Дуги множини N з подальшого розгляду вилучаються, оскільки в них зміни потоку неможливі. Зафарбовуємо вершину s (джерело).

Крок 2. Зафарбовуємо дуги й вершини відповідно до наведених нижче правил, поки або не буде пофарбовано вершину t, або забарвлення нових вершин стане неможливим.

Правила забарвлення вершини y й дуги (x,y), коли вершину x вже пофарбовано, можна сформулювати таким чином:

– якщо $(x,y) \in I$, то фарбують вершину y й дугу (x,y);

– коли $(y,x) \in R$, то фарбують вершину y і дугу (y,x);

– в інших випадках забарвлення вершини y й відповідної дуги не відбувається.

Алгоритм описано.

Приклад 4. Знайти збільшувальний ланцюг у графі, поданому на рис. 2.27. Поруч із дугами графа зазначено букви R, I та N, які встановлюють їх належність до відповідних множин.

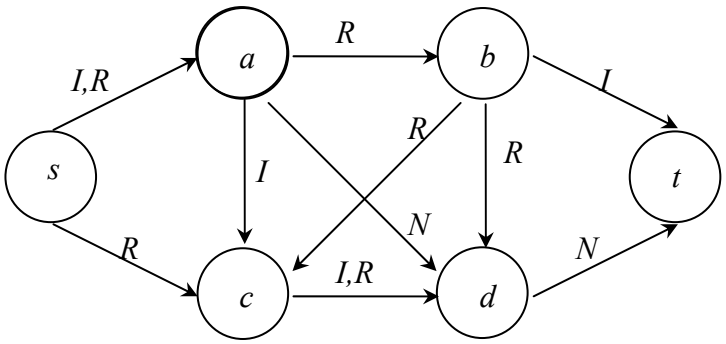


Рис. 2.27. Граф до прикладу 4

Розв'язування

Крок 1. Дуги (a,d) й (d,t) належать множині N, тому вони з подальшого розгляду вилучаються.

Крок 2. Насамперед зафарбовується вершина s. Від неї можуть бути пофарбовані вершини a й дуга (s,a), оскільки $(s,a) \in I$. Вершина c й дуга (s,c) не можуть бути пофарбовані від вершини s, тому що $(s,c) \in R$. Отже, поточне дерево забарвлених дуг має вигляд (дуги позначено товстою лінією), показаний на рис. 2.28.

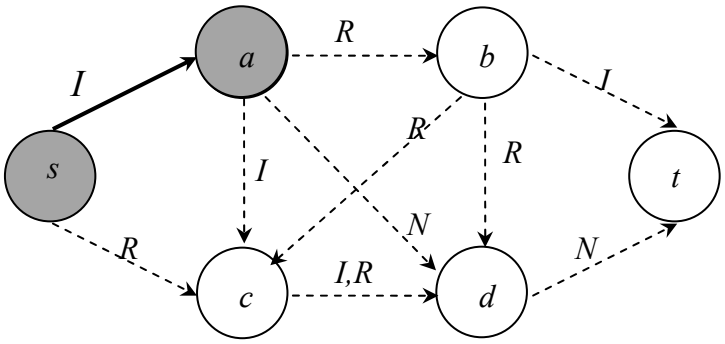


Рис. 2.28. Перший етап побудови збільшувального ланцюга у графі

Далі будемо зафарбовувати вершини й дуги від вершини a. Вершина b й дуга (a,b) не можуть бути пофарбовані, оскільки $(a,b)\in R$. Вершина c й дуга (a,c) можуть бути пофарбовані, тому що $(a,c)\in I$. Це завершує процедуру забарвлення елементів графа від вершини a.

Поточне дерево пофарбованих дуг набуває вигляду, показаного на рис. 2.29.

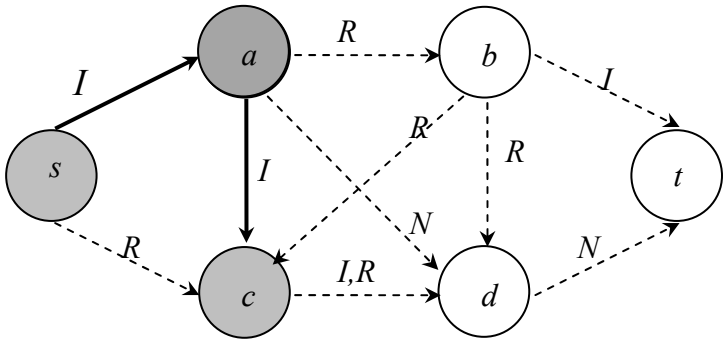


Рис. 2.29. Другий крок побудови збільшувального ланцюга у графі

Далі будемо зафарбовувати вершини й дуги від вершини c. Оскільки вершини s й a уже пофарбовані, їх можна не розглядати.

Вершина b й дуга (b,c) можуть бути пофарбовані, тому що $(b,c)\in R$. Так само можуть бути пофарбовані d й дуга(c,d), оскільки $(c,d)\in I$. Це завершує процедуру фарбування від вершини c й поточне дерево забарвлених дуг має вигляд, показаний на рис. 2.30.

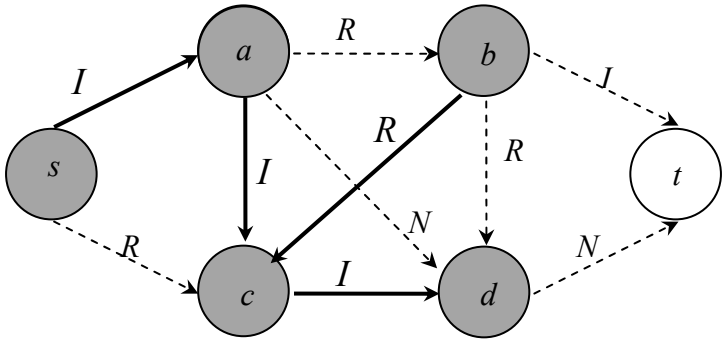


Рис. 2.30. Третій крок побудови збільшувального ланцюга у графі

Далі будемо зафарбовувати вершини й дуги від вершини b. Вершини a,c й d, які вже були пофарбовані, можна не розглядати. Вершина t й дуга (b,t) можуть бути пофарбовані, тому що $(b,t)\in I$.

На цьому процедура забарвлення закінчується, оскільки виявилася пофарбованою вершина t.

Дерево, яке складається з пофарбованих вершин і дуг вихідного графа, показано на рис. 2.31.

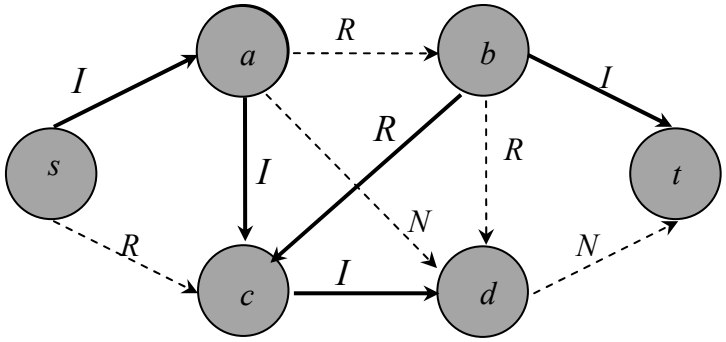
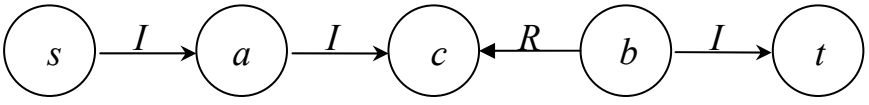


Рис. 2.31. Дерево пофарбованих вершин і дуг з прикладу 4

На рис. 2.31. бачимо, що збільшувальним ланцюгом з вершини s до вершини t є такий:



Нехай $i(s,a)=4,\quad i(a,c)=3,\quad r(b,c)=2,\quad i(b,t)=2$. Тоді максимальне збільшення потоку по цьому ланцюгу має таке значення:

$$\min\{i(s,a),i(a,c),r(b,c),i(b,t)\}=\min\{4,3,2,2\}=2.$$

Таким чином, потоки в прямих дугах знайденого ланцюга, тобто в дугах $(s,a),(a,c)$ і (b,t) можуть бути збільшені на 2 одиниці. Останнє свідчить про те, що зазначені 2 одиниці, які раніше проходили по дузі (b,c), «повертаються» по дузі (b,t), а відповідна «нестача потоку» у вершині c компенсується за рахунок двох одиниць, пропущених по дузі (a,c).

Щоразу, коли в процесі виконання розробленого алгоритму визначають збільшувальний ланцюг від вершини s до вершини t, із джерела в стік може бути відправлено додаткову кількість одиниць потоку, і це не перевищує максимальної величини потоку по даному ланцюгу. При цьому в збільшувальних дугах знайденого ланцюга потік зростає, а в зменшувальних дугах спадає на зазначену максимальну величину.

2.4.2. Алгоритм пошуку максимального потоку (Форда й Фалкерсона)

Ідея цього алгоритму полягає в тому, що береться початковий потік від вершини s до вершини t й за допомогою алгоритму пошуку збільшувального ланцюга, виконується його пошук. Якщо він виявляється успішним, то потік уздовж знайденого ланцюга збільшується до максимально можливого значення. Потім здійснюють пошук нового збільшувального ланцюга, і т. д. Коли ж на якомусь етапі цієї процедури ланцюг, що збільшує потік, знайти не вдається, то це означає, що потік від вершини s до вершини t є максимальним.

Опишемо алгоритм пошуку максимального потоку.

Крок 1. Вибрати будь-який початковий потік від вершини s (джерело) мережі A до вершини t (стік), тобто будь-який набір величин $f(x,y)$, що задовольняє такі співвідношення:

– кількість одиниць, які надходять у вершину x, дорівнює кількості одиниць які виходять з цієї вершини (x – множина вершин мережі A), тобто

$$\sum_{y\in x} f(x,y) - \sum_{y\in x} f(y,x) = 0, \qquad (x \neq s,\ x \neq t);$$

– кількість одиниць потоку, що проходить по дузі, не перевищує пропускну здатність дуги, а саме:

$$0 \le f(x,y) \le c(x,y), \quad (x,y) \in A;$$

– сумарне число одиниць потоку, що виходить із джерела, повинно дорівнювати сумарному числу одиниць (V), що потрапляють у стік, тобто

$$\sum_{y\in x} f(s,y) - \sum_{y\in x} f(y,s) = V,$$
$$\sum_{y\in x} f(y,t) - \sum_{y\in x} f(t,y) = V.$$

Якщо величина жодного з початкових потоків від вершини s до вершини t невідома, то можна її задати, прийнявши для всіх дуг $(x,y)\in A$, що $f(x,y)=0$.

Крок 2. Кожну з дуг (x,y) графа віднести до однієї з множин I або R за таким правилом:

Коли $f(x,y)<c(x,y)$, то задати, що $i(x,y)=c(x,y)-f(x,y)$, і вважати, що дуга (x,y) належить множині I.

Коли $f(x,y)>0$, то прийняти, що $r(x,y)=f(x,y)$, і вважати, що дуга (x,y) належить множині R.

Крок 3. На множинах I і R, сформованих у попередньому кроці, застосувати до вихідного графа алгоритм пошуку збільшувального ланцюга. Якщо при цьому такий ланцюг знайти не вдається, то закінчити процедуру виконання алгоритму: поточний потік є максимальним. А якщо ні, то здійснити максимально можливе збільшення потоку вздовж знайденого збільшувального ланцюга і повернутися до кроку 2.

Алгоритм описано.

Приклад 5. Застосуємо алгоритм пошуку максимального потоку до графа, зображеного на рис. 2.32. Величини $c(x,y)$ позначено біля дуг.

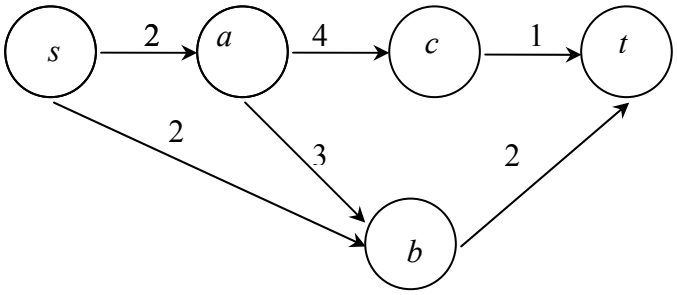


Рис. 2.32. Граф до прикладу 5

Розв'язування

Крок 1. Виконання алгоритму починається із задання нульового потоку, тобто для всіх дуг (x,y) графа вважаємо, що $f(x,y)=0$.

Крок 2. Оскільки для всіх дуг виконується умова, що $f(x,y)<c(x,y)$, то всі вони можуть будуть віднесені до множини I. При цьому $i(x,y)=c(x,y)-f(x,y)=c(x,y)$. Крім того, враховуючи, що для всіх дуг

$f(x,y)=0$, то на кожному кроці жодну з них не можна віднести до множини R, тобто маємо граф, зображений на рис. 2.33.

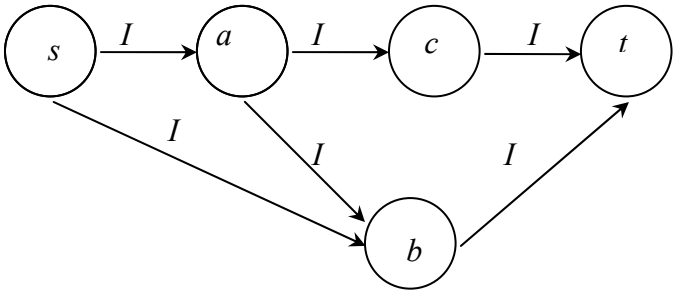


Рис. 2.33. Перший етап пошуку максимального потоку (приклад 5)

При цьому $i(s,a)=c(s,a)=2;$

$i(a,c)=c(a,c)=4;$

$i(c,t)=c(c,t)=1;$

$i(s,b)=c(s,b)=2;$

$i(b,t)=c(b,t)=2;$

$i(a,b)=c(a,b)=3.$

Крок 3. Скористаємося алгоритмом пошуку збільшувального ланцюга від вершини s до вершини t, який було описано вище.

На даному етапі виконання алгоритму в графі існує кілька таких ланцюгів (оскільки початковий потік нульовий). Їх показано на рис. 2.34.

Розглянемо один з них, а саме, ланцюг $(s,a),(a,b),(b,t)$, зображений на рис. 2.34, *б*.

Максимальне збільшення потоку на цьому ланцюзі обчислюється таким чином:

$$\min\{i(s,a),i(a,b),i(b,t)\}=\min\{2,3,2\}=2.$$

Отже, потік у кожній із трьох його дуг збільшується на дві одиниці, в інших дугах потік не змінюється, тобто

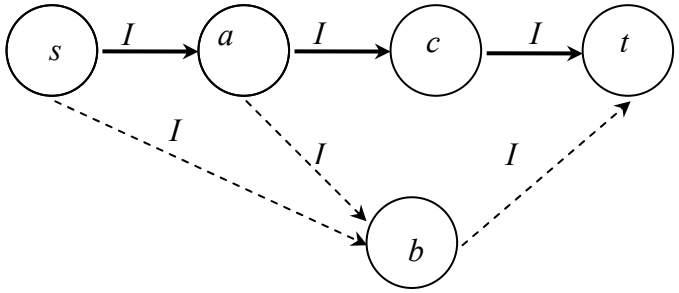
$$f(s,a)=2,\quad f(a,b)=2,\quad f(b,t)=2,\quad f(s,b)=0,\quad f(a,c)=0,\quad f(c,t)=0.$$

Крок 2. Обчислюємо знову величини $i(x,y)$ й $r(x,y)$ за формулами (2.1), (2.2) і коректуємо склад множин I і R, а саме:

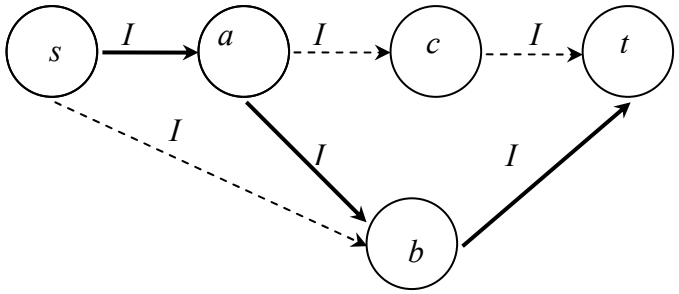
$$i(x,y)=c(x,y)-f(x,y);$$

$$r(x,y)=f(x,y).$$

а



б



в

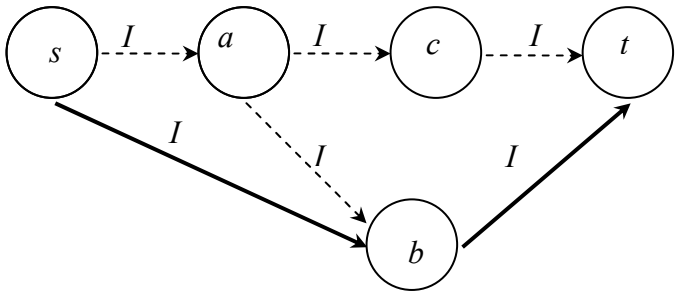


Рис. 2.34. Результат застосування алгоритму пошуку збільшувального ланцюга для графа з прикладу 5 (перший прохід алгоритму)

Оскільки $f(s,a)=2=c(s,a)$, а отже $i(s,a)=0$ і тому дуга $(s,a)\notin I$, $r(s,a)=2$, це означає, що дуга $(s,a)\in R$.

Через те, що $f(a,b)=2<c(a,b)=3$, дуга $(a,b)\in I$, а значення $i(a,b)=1$, крім того $(a,b)\in R$, а значення $r(a,b)=2$.

Аналогічно для дуги (b,t) отримуємо такі результати:

$f(b,t)=2=c(b,t)$, тому $(b,t)\notin I$, оскільки $r(b,t)=2$, то $(b,t)\in R$.

Дуги$(s,b),(a,c),(c,t)$, як і раніше, належать до множини I.

Надпишемо символи I й R над відповідними дугами графа (див. рис. 2.35).

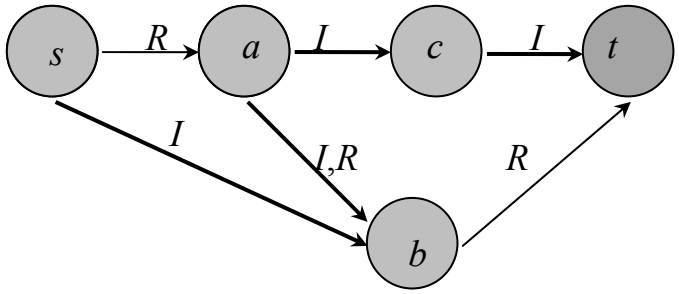


Рис. 2.35. Ілюстрація до розв'язування прикладу 5

Крок 3. Знову скористаємося алгоритмом пошуку збільшувального ланцюга, від вершини s до вершини t. У цьому випадку (див. рис. 2.35) такий ланцюг єдиний. Його зображено на рис. 2.36

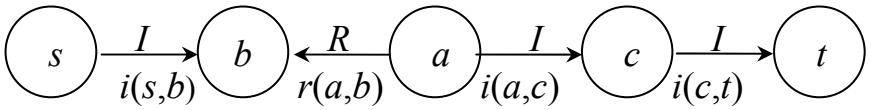


Рис. 2.36. Збільшувальний ланцюг до прикладу 5 (другий прохід алгоритму)

Максимальне збільшення потоку вздовж ланцюга, поданого на рис. 2.35, набуває такого значення:

$$\min\{i(s,b),r(a,b),i(a,c),i(c,t)\}=\min\{3,2,4,1\}=1.$$

Таким чином, від вершини s до вершини t може бути додатково пропущено одну одиницю потоку. При цьому величина потоку в кожній із трьох прямих дуг$(s,b),(a,c)(c,t)$, що належать знайденому ланцюгу, на одиницю збільшиться, а значення потоку в зворотній дузі (a,b) на одиницю зменшиться.

Отже, тепер потоки в дугах розглянутого графа будуть мати такі значення:

$$f(s,a)=2,\quad f(a,b)=2,\quad f(b,t)=2,$$
$$f(s,b)=1,\quad f(a,c)=1,\quad f(c,t)=1.$$

Кожна із трьох одиниць побудованого на даний момент потоку проходить такими маршрутами:

– перша одиниця проходить від вершини s до вершини t таким шляхом: $(s,a),(a,b)$ і (b,t);

– друга одиниця проходить від вершини s до вершини t по такому шляху: $(s,b),(b,t)$;

– третя одиниця проходить від вершини s до вершини t по такому шляху: $(s,a),(a,c)$ і (c,t).

Крок 2. Обчислюємо знову величини $i(x,y)$ й $r(x,y)$ та коректуємо склад множин I і R, а саме:

$$f(a,b)=1<c(a,b)=3,\ (a,b)\in I,\ i(a,b)=2;\ (a,b)\in R,\ r(a,b)=1;$$
$$f(s,b)=1<c(s,b)=3,\ (s,b)\in I,\ i(s,b)=2;\ (s,b)\in R,\ r(s,b)=1;$$
$$f(a,c)=1<c(a,c)=4,\ (a,c)\in I,\ i(a,c)=3;\ (a,c)\in R,\ r(a,c)=1;$$
$$f(c,t)=1<c(c,t)=1,\ (c,t)\notin I,\ r(c,t)=1;\ (c,t)\in R.$$

Висновки стосовно дуг (s,a) і (b,t) зроблено вище.

Проставимо на графі символи I й R, які отримано для інших дуг (рис. 2.37).

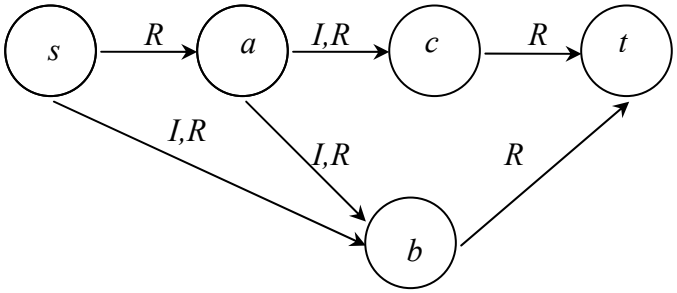


Рис. 2.37. Пошук збільшувального ланцюга в прикладі 5 (третій прохід алгоритму)

Крок 3. Знову скористаємося алгоритмом пошуку збільшувального ланцюга, від вершини s до вершини t.

Для цього проаналізуємо граф, поданий на рис. 2.37. Виявляється, що з огляду на знайдені значення потоку в дугах графа, він не має жодного збільшувального ланцюга. Під час виконання алгоритму у вихідному графі (рис. 2.37.) виявляються забарвленими вершини s, b, a і c (у зазначеному порядку), однак вершина t не зафарбовується. Отже, оскільки збільшувального ланцюга тут не існує, то виконання алгоритму пошуку максимального потоку завершується.

Останній із побудованих потоків є максимальним, бо від вершини s до вершини t можна пропустити 3 одиниці потоку.

Питання для самоконтролю

1. Дайте визначення графа.
2. Якими способами можна задати граф?
3. Яка властивість графа називається ізоморфізмом?
4. Дайте визначення матриці суміжності графа.
5. У якому випадку матриця суміжності графа буде мати на своїй головній діагоналі відмінні від нуля елементи?
6. Чим визначається розмір матриці інциденцій графа?
7. Що являє собою орієнтоване дерево найкоротших шляхів графа?
8. Що є ознакою завершення виконання алгоритму побудови найкоротшого кістякового дерева?
9. Які дуги в задачі про максимальний потік у графі називаються збільшувальними? Зменшувальними?
10. Чи може збільшувальний шлях у задачі про максимальний потік містити зменшувальні дуги?

Задачі для самостійного розв'язування

1. Скласти матриці суміжності й інциденцій для заданих нижче неорієнтованих графів:

а)
$X=\{x_1,x_2,x_3,x_4\}$; $G(x_1)=\{x_2,x_4\}$; $G(x_2)=\{x_1,x_3,x_4\}$; $G(x_3)=\{x_2,x_3\}$;
$G(x_4)=\{x_1,x_2,x_3\}$.

б)
$X=\{x_1,x_2,x_3,x_4\}$; $G(x_1)=\{x_1,x_4\}$; $G(x_2)=\{x_3,x_4\}$; $G(x_3)=\{x_2,x_3,x_4\}$;
$G(x_4)=\{x_1,x_2,x_3\}$.

2. Зобразити графічно неорієнтований граф, заданий такою матрицею суміжності:

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

3. Задати аналітично орієнтований граф, зображений нижче.

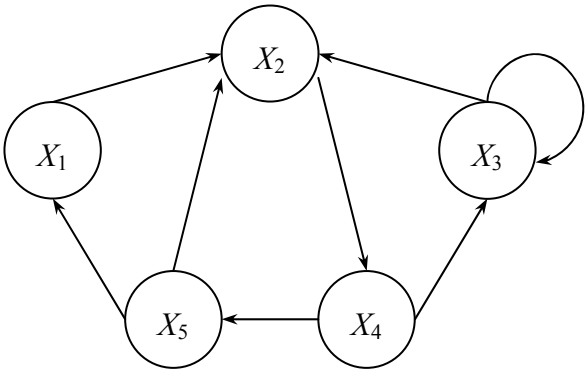


Рис. 2.38. Граф до задачі 3

4. Зобразити графічно орієнтований граф, заданий такою матрицею інциденцій:

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

5. Знайти найкоротший шлях від вершини s до вершини t за допомогою алгоритму Дейкстри для заданого нижче графа.

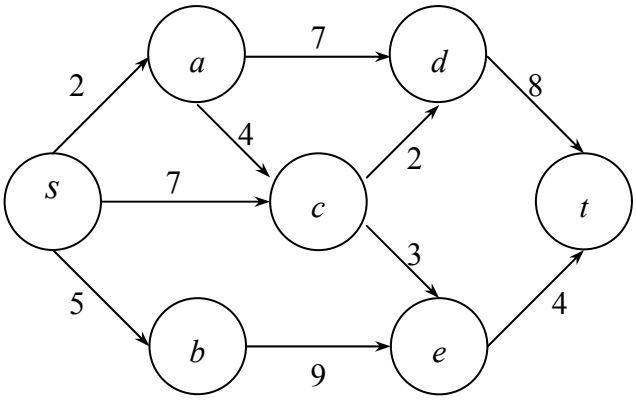


Рис. 2.39. Граф до задачі 5

6. Для заданого неорієнтованого графа (рис. 2.40) побудувати найкоротше кістякове дерево.

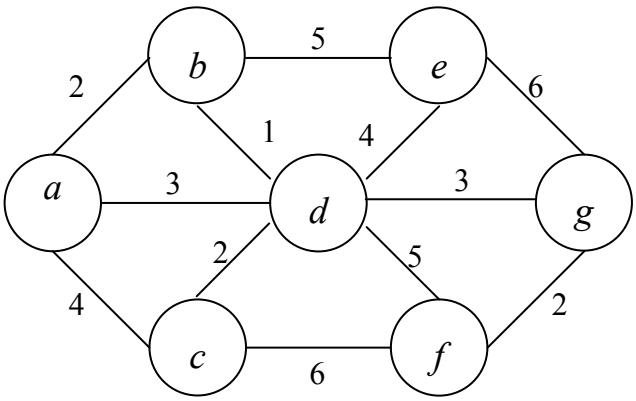


Рис. 2.40. Граф до задачі 6

7. Знайти збільшувальний шлях від джерела s до стоку t для зображеного нижче графа.

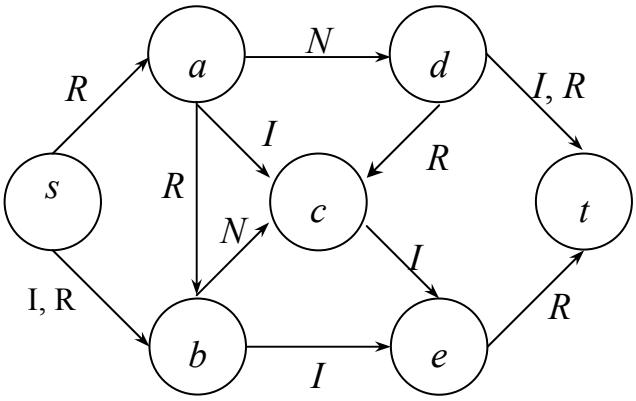


Рис. 2.41. Граф до задачі 7

8. Знайти максимальну величину потоку від вершини s до вершини t, використовуючи алгоритм Форда й Фалкерсона. На дугах графа (рис. 2.42) зазначено їхні максимальні пропускні можливості.

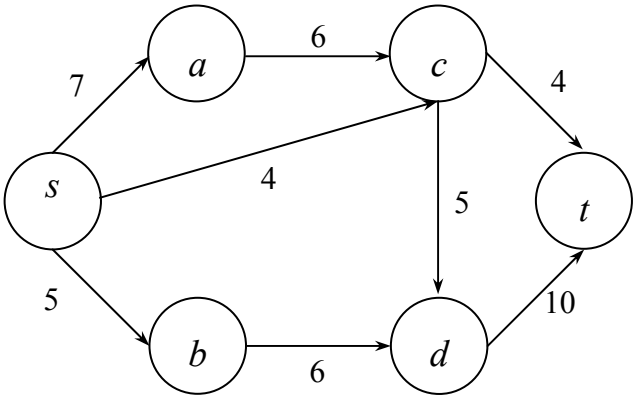


Рис. 2.42. Граф до задачі 8

9. Для графа, поданого на рис. 2.43, за допомогою алгоритму Дейкстри побудувати дерево найкоротших шляхів із коренем у вершині s.

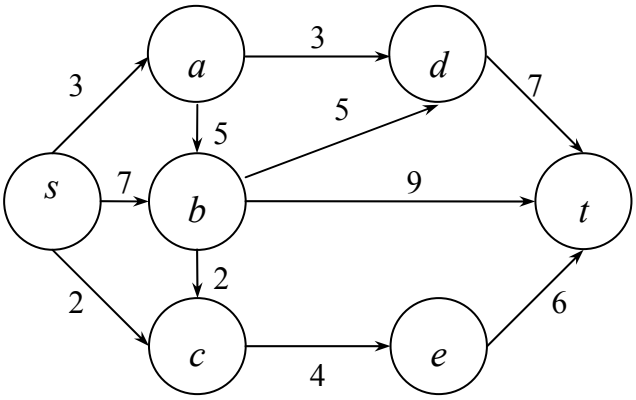


Рис. 2.43. Граф до задачі 9

10. Скласти аналітичний опис неорієнтованого графа за такою матрицею суміжності:

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

РОЗДІЛ 3

ЕЛЕМЕНТИ МАТЕМАТИЧНОЇ ЛОГІКИ Й ТЕОРІЇ АВТОМАТІВ

Мета розділу: *вивчення основних понять математичної логіки й набуття практичних навичок мінімізації логічних функцій і синтезу скінченних автоматів*

§ 3.1. Основні поняття

Двозначна логіка дозволяє математично описати реальні об'єкти, які перебувають в одному із двох можливих станів. Такі об'єкти описуються за допомогою *булевих змінних*, кожна з яких має лише два можливі значення – нуль або одиницю.

Відношення між булевими змінними задають за допомогою *булевих функцій*, які, подібно до числових функцій, можуть залежати в загальному випадку від кількох булевих змінних, тобто $y = f\left(x_1, x_2, ..., x_n\right)$.

Найважливішою властивістю булевих функцій є те, що вони, як і булеві змінні, можуть мати тільки два можливі значення – нуль або одиницю. Ця властивість дозволяє використовувати їх як булеві змінні в інших функціях.

Довільна булева функція задається одним з трьох способів: матричним (табличним), геометричним й аналітичним. При матричному способі булева функція $f\left(x_1, x_2, ..., x_n\right)$ задається за допомогою *таблиці істинності* (див. табл. 3.1 і 3.2), у лівій частині якої подано всі можливі двійкові набори довжини n, а у правій наведено значення функції, що відповідають цим наборам.

Під *двійковим набором*: $y = y_1, y_2, ..., y_n$ (тут y_i набуває значень 0 або 1, для всіх значень індекса i), розуміється сукупність значень аргументів $x_1, x_2, ..., x_n$ булевої функції f. Двійковий набір має довжину n, якщо він включає n цифр із множини $\{0,1\}$. Наприклад, у табл. 3.1 подано всі двійкові набори довжини 3.

Іноді двійкові набори в таблиці істинності булевої функції зручно записувати, використовуючи *номери наборів*. Для цього запишемо аргументи $x_1, x_2, ..., x_n$ в порядку зростання їх індексів. Тоді будь-який двійковий набір: $y = y_1, y_2, ..., y_n$, $y_i \in \{0,1\}$, можна розглядати як ціле двійкове число N, а саме: $N = y_1 \cdot 2^{n-1} + y_2 \cdot 2^{n-2} + \ldots + y_n$. Це число називають *номером набору* y. Наприклад, двійкові набори 101 і 111 мають номери 5 і 7 відповідно. Очевидно, що будь-яка булева функція може бути задана таблицею істинності, у якій двійкові набори замінено на їхні номери (табл. 3.2).

Таблиця 3.1

$x_1\, x_2\, x_3$	$f(x_1, x_2, x_3)$
000	0
001	1
010	0
011	0
100	1
101	1
110	0
111	1

Таблиця 3.2

номери наборів	$f(x_1, x_2, x_3)$
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	1

Булеві функції, що залежать від великого числа змінних, задавати таблицею істинності незручно через їхню громіздкість. Наприклад, таблиця істинності булевої функції 8 змінних буде містити 2^8, тобто 256 рядків. У зв'язку з цим для задання функцій багатьох змінних зручно використовувати модифікацію таблиці істинності.

Розглянемо спосіб її побудови для функції n змінних. Множина n змінних функції розбивається на дві підмножини: $x_1, x_2, ..., x_{j-1}$ і $x_j, x_{j+1}, x_{j+2}, ..., x_n$. Змінними $x_1, x_2, ..., x_{j-1}$ позначають рядки таблиці істинності, записуючи в кожному з них відповідні двійкові набори довжини j–1. Змінними x_j, $x_{j+1}, x_{j+2}, ..., x_n$ відмічають стовпчики таблиці, записуючи в кожному з них відповідні двійкові набори довжини $n - j + 1$. Значення функції записується в клітинці на перетині відповідного рядка й стовпця (табл. 3.3).

Таблиця 3.3

$x_1, x_2, ..., x_{j-1}$	$x_j, x_{j+1}, x_{j+2}, ..., x_n$.			
	00…0	00…1	…	11…1
00…0			…	
00…1			…	
…	…	…	…	…
11…1			…	

Основними в булевій алгебрі виступають три функції:

1. *Заперечення* (позначається як $y = \overline{x}$).

Це функція однієї змінної, що набуває протилежного (інверсного) стосовно неї значення, тобто

$$y = \bar{x} = \begin{cases} 0, & \text{якщо } x = 1, \\ 1, & \text{якщо } x = 0. \end{cases}$$

2. *Диз'юнкція* (логічне додавання).

Це функція двох змінних x_1 й x_2, яка набуває значення «0», тільки тоді, коли $x_1 = 0$ й $x_2 = 0$ одночасно, тобто

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Диз'юнкція позначається таким чином: $y = x_1 \vee x_2$ або $y = x_1 + x_2$.

3. *Кон'юнкція* (логічне множення).

Це функція двох змінних x_1 і x_2, яка набуває значення «1» тільки тоді, коли $x_1 = 1$ і $x_2 = 1$ одночасно, тобто

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

Кон'юнкція позначається як $y = x_1 \cdot x_2$ або $y = x_1 \wedge x_2$.

Вирази $\bar{x}$, $x_1 \vee x_2$, $x_1 \wedge x_2$ являють собою *логічні формули*. Більш складні формули створюють шляхом заміщення змінних іншими формулами, які звичайно беруться в дужки. Наприклад: $(x_1 \vee x_2) \vee \overline{(x_1 \wedge x_3)}$. Кожна формула визначає деяку булеву функцію. Це *аналітичний* спосіб задання функції.

Дві функції (і відповідні формули) є *рівносильними*, якщо при будь-яких значеннях аргументів вони набувають однакових значень. Рівносильні формули з'єднуються знаком «дорівнює» (=).

Універсальний спосіб перевірки тотожності функцій полягає у використанні таблиць істинності.

Приклад 6: Перевіримо тотожність таких функцій: $y_1 = \overline{x_1 \vee x_2}$ і $y_2 = \bar{x}_1 \wedge \bar{x}_2$.

Складемо таблиці істинності. Для цього обчислимо значення функцій y_1 і y_2 для будь-яких можливих комбінацій значень змінних x_1 і x_2:

x_1	x_2	y_1	y_2
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Оскільки для одних і тих самих значень змінних функції y_1 й y_2 набувають однакових значень, то вони рівносильні.

Наведемо основні тотожності булевої алгебри, які неважко довести за допомогою таблиць істинності.

1. *Комутативність* логічного додавання й множення, тобто

$$x \vee y = y \vee x;$$
$$x \wedge y = y \wedge x.$$

2. *Асоціативність* логічного додавання й множення

$$x \vee (y \vee z) = (x \vee y) \vee z;$$
$$x \wedge (y \wedge z) = (x \wedge y) \wedge z.$$

3. *Дистрибутивність* множення відносно додавання й додавання відносно множення:

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z);$$
$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z).$$

4. Властивості констант:

$$x \vee 0 = x \quad x \vee 1 = 1;$$
$$x \wedge 0 = 0 \quad x \wedge 1 = x.$$

5. Властивості заперечення:

$$x \vee \bar{x} = 1;$$
$$x \wedge \bar{x} = 0.$$

6. Закони де Моргана:

$$\overline{x \vee y} = \bar{x} \wedge \bar{y};$$
$$\overline{x \wedge y} = \bar{x} \vee \bar{y}.$$

7. Закони поглинання:

$$x \vee (x \wedge y) = x;$$
$$x \wedge (x \vee y) = x.$$

8. Закони ідемпотентності:

$$x \vee x = x;$$
$$x \wedge x = x.$$

§ 3.2. Мінімізація логічних функцій

Із положень **§** 3.1 випливає, що одна й та сама логічна функція може мати різну форму запису. Особливо виділяються диз'юнктивна й кон'юнктивна нормальні форми (ДНФ і КНФ).

Диз'юнктивною нормальною формою (ДНФ) називається диз'юнкція скінченного числа різних членів, кожний з яких являє собою кон'юнкцію окремих змінних або їх заперечень, що входять у даний член не більше одного разу.

Кон'юнктивною нормальною формою (КНФ) називається кон'юнкція скінченного числа різних членів, кожний з яких являє собою диз'юнкцію окремих змінних або їх заперечень, котрі входять у даний член не більше одного разу.

Використовуючи наведені вище закони, будь-яку формулу можна звести до ДНФ або до КНФ.

Приклад 7. Запишемо функцію $\left(x_1 \cdot x_2 \vee \overline{x_2} x_3\right) \cdot \overline{\overline{x_1} \cdot x_4}$ у вигляді ДНФ і КНФ.

Спочатку перетворимо вираз $\overline{\overline{x_1} \cdot x_4}$. Використовуючи закони де Моргана й властивості заперечення, отримуємо, що $\overline{\overline{x_1} \cdot x_4} = \overline{\overline{x_1}} \vee \overline{x_4} = x_1 \vee \overline{x_4}$.

Тепер перетворимо вихідну функцію, застосувавши дистрибутивний закон множення відносно додавання та закони поглинання, а саме:

$$\left(x_1 \cdot x_2 \vee \overline{x_2} x_3\right) \cdot \overline{\overline{x_1} \cdot x_4} = \left(x_1 \cdot x_2 \vee \overline{x_2} x_3\right) \cdot \left(x_1 \vee \overline{x_4}\right) = x_1 x_2 \vee x_1 \overline{x_2} x_3 \vee x_1 x_2 \overline{x_4} \vee \overline{x_2} x_3 \overline{x_4}.$$

Отже, ДНФ даної функції має такий вигляд: $x_1 x_2 \vee x_1 \overline{x_2} x_3 \vee x_1 x_2 \overline{x_4} \vee \overline{x_2} x_3 \overline{x_4}$.

Виконаємо перетворення вихідної функції за допомогою другого дистрибутивного закону, таким чином:

$$\left(x_1 \cdot x_2 \vee \overline{x_2} x_3\right) \cdot \overline{\overline{x_1} \cdot x_4} = \left(x_1 \cdot x_2 \vee \overline{x_2} x_3\right) \cdot \left(x_1 \vee \overline{x_4}\right) = \left(x_1 \vee \overline{x_2} x_3\right)\left(x_2 \vee \overline{x_2} x_3\right) \cdot \left(x_1 \vee \overline{x_4}\right) =$$
$$= \left(x_1 \vee \overline{x_2}\right)\left(x_1 \vee x_3\right)\left(x_2 \vee \overline{x_2}\right) \times \left(x_2 \vee x_3\right)\left(x_1 \vee \overline{x_4}\right) =$$
$$= \left(x_1 \vee \overline{x_2}\right)\left(x_1 \vee x_3\right)\left(x_2 \vee x_3\right)\left(x_1 \vee \overline{x_4}\right).$$

Останнє перетворення отримано з огляду на властивості заперечення.

Таким чином, КНФ даної функції має такий вигляд:

$$\left(x_1 \vee \overline{x_2}\right)\left(x_1 \vee x_3\right)\left(x_2 \vee x_3\right)\left(x_1 \vee \overline{x_4}\right).$$

Члени диз'юнктивної нормальної форми, що включають k змінних називають *мінтермами* k-го рангу.

Члени кон'юнктивної нормальної форми, складені з k змінних, називають *макстермами* k-го рангу.

Наприклад: $x_1 x_2$ – мінтерм 2-го рангу; $x_1 \overline{x_2} x_3$ – мінтерм 3-го рангу; $x_1 \vee \overline{x_2}$ – макстерм 2-го рангу.

Якщо в кожному члені нормальної форми містяться всі змінні (або в прямому, або в інверсному вигляді), то вона називається *досконалою* (диз'юнктивною або кон'юнктивною) *нормальною* формою і позначається як ДДНФ або ДКНФ відповідно. Перехід від ДНФ (КНФ) до досконалої форми виконується на основі такого співвідношення: $x \vee \overline{x} = 1\left(x \wedge \overline{x} = 0\right)$.

Приклад 8. Записати функцію: $x_1 x_2 \vee x_1 \overline{x_2} x_3 \vee x_1 x_2 \overline{x_4} \vee \overline{x_2} x_3 \overline{x_4}$ у вигляді досконалої форми.

Розв'язування

$$x_1 x_2 \vee x_1 \overline{x_2} x_3 \vee x_1 x_2 \overline{x_4} \vee \overline{x_2} x_3 \overline{x_4} = x_1 x_2 \cdot 1 \cdot 1 \vee x_1 \overline{x_2} x_3 \cdot 1 \vee x_1 x_2 \cdot 1 \cdot \overline{x_4} \vee 1 \cdot \overline{x_2} x_3 \overline{x_4} =$$
$$= x_1 x_2 \left(\overline{x_3} \vee x_3\right)\left(\overline{x_4} \vee x_4\right) \vee x_1 \overline{x_2} x_3 \left(x_4 \vee \overline{x_4}\right) \vee x_1 x_2 \left(x_3 \vee \overline{x_3}\right) \cdot \overline{x_4} \vee \left(x_1 \vee \overline{x_1}\right) \overline{x_2} x_3 \overline{x_4} =$$
$$= x_1 x_2 x_3 x_4 \vee x_1 x_2 x_3 \overline{x_4} \vee x_1 x_2 \overline{x_3} x_4 \vee x_1 x_2 \overline{x_3}\,\overline{x_4} \vee x_1 \overline{x_2} x_3 x_4 \vee$$
$$\vee x_1 \overline{x_2} x_3 \overline{x_4} \vee x_1 x_2 x_3 \overline{x_4} \vee x_1 x_2 \overline{x_3}\,\overline{x_4} \vee x_1 \overline{x_2} x_3 \overline{x_4} \vee \overline{x_1}\,\overline{x_2} x_3 \overline{x_4} =$$
$$= x_1 x_2 x_3 x_4 \vee x_1 x_2 x_3 \overline{x_4} \vee x_1 x_2 \overline{x_3} x_4 \vee x_1 x_2 \overline{x_3}\,\overline{x_4} \vee x_1 \overline{x_2} x_3 x_4 \vee x_1 \overline{x_2} x_3 \overline{x_4} \vee \overline{x_1} x_2 x_3 \overline{x_4}.$$

ДКНФ має такий вигляд:

$$x_1 x_2 x_3 x_4 \vee x_1 x_2 x_3 \overline{x_4} \vee x_1 x_2 \overline{x_3} x_4 \vee x_1 x_2 \overline{x_3}\,\overline{x_4} \vee x_1 \overline{x_2} x_3 x_4 \vee x_1 \overline{x_2} x_3 \overline{x_4} \vee \overline{x_1}\,\overline{x_2} x_3 \overline{x_4}.$$

Оскільки булева функція може бути записана по-різному, виникає потреба пошуку найбільш компактної форми її подання. Ця процедура називається *мінімізацією логічних функцій*. Для розв'язування задачі мінімізації логічна функція попередньо зводиться до ДДНФ або ДКНФ.

Мінімізація функцій у булевому базисі відбувається згідно із законами поглинання й зводиться до пошуку мінімальної диз'юнктивної або кон'юнктивної форми. За критерій звичайно беруть змінну C – ціну покриття, що обчислюється за таким виразом:

$$C = \sum_{s} g_s \left(n - S \right),$$

де g_s – число термів рангу S, що утворюють покриття функції від n змінних.

Найпоширенішими методами мінімізації є методи Квайна, Квайна – Мак-Класкі та Вейча – Карно.

Вихідною формою для кожного із цих методів виступає одна з досконалих форм: ДДНФ або ДКНФ.

У загальному випадку кожний з вищезазначених методів передбачає трьохєтапний алгоритм мінімізації, а саме:

Етап 1. Проводиться перехід від *досконалої* Д(К)НФ до *скороченої* Д(К)НФ шляхом об'єднання спочатку конституент, а потім усіх похідних членів більш низького рангу.

Скороченою формою називається Д(К)НФ, членами якої служать тільки ізольовані елементарні кон’юнкції (диз'юнкції).

Члени скороченої Д(К)НФ в алгебрі логіки називаються *первинними імплікантами* (*імпліцентами*).

Етап 2. Проводиться перехід від *досконалої* Д(К)НФ до *тупикової* Д(К)НФ.

Тупиковою називається Д(К)НФ, членами якої є прості імпліканти (імпліценти), причому серед них немає жодної зайвої.

Зайвим називається такий член функції, вилучення якого не впливає на значення істинності цієї функції. Назва «тупикова» показує, що подальша мінімізація функції в рамках нормальних форм уже неможлива.

Етап 3. Виконується перехід від *тупикової* (мінімальної серед нормальних) форми функції до її *мінімальної* форми.

Цей етап, називаний звичайно *факторизацією*, уже не є регулярним, як два попередні, а вимагає певної вправності, інтуїції й досвіду.

Іншими словами пошук можливостей спрощення функції здійснюється шляхом спроб і випробувань. Для зменшення числа операцій заперечення слід застосовувати закон інверсії, а для зменшення числа кон’юнкцій і диз'юнкцій – відповідні розподільні закони.

3.2.1. Метод Квайна

Ідея мінімізації логічної функції за методом Квайна полягає в попарному порівнянні всіх імплікант, які входять до складу ДДНФ, з метою виявлення можливості поглинання якоїсь змінної (ця операція називається склеюванням)

Опишемо алгоритм цього методу.

Крок 1. *Перетворення логічної функції у* ДДНФ.

Якщо функція містить диз'юнктивні члени, що не належать до мінтермів, то їх необхідно розгорнути в мінтерми.

Крок 2. *Знаходження первинних імплікант*.

Кожний мінтерм ДДНФ порівнюється з основними мінтермами цієї форми. Щоб при порівнянні не пропутити жодного мінтерма, операцію склеювання бажано робити за допомогою спеціальних таблиць, у верхньому рядку й лівому стовпчику яких розташовуються всі мінтерми, що брали участь у порівнянні. Якщо якісь мінтерми відрізняються меж собою тільки однією змінною (тобто в одному це x, а в іншому $\bar{x}$), то в клітинці таблиці, розташованої проти склеюваних мінтермів, виписується тільки їхня спільна частина. Заміна двох мінтермів їх спільною частиною (склеювання) еквівалентна такій операції: $XY + \bar{X}Y = Y$. Процес склеювання за допомогою таблиць триває до тих пір, поки не буде одержано терми, склеювання яких неможливе.

На цьому процедура складання скороченої ДНФ закінчується.

Крок 3. *Розміщення позначок.*

Складається таблиця, число рядків якої дорівнює кількості отриманих первинних імплікант, а число стовпців збігається із кількістю мінтермів ДДНФ. Якщо в деякий мінтерм ДДНФ входить одна з первинних імплікант, то в місці перетину відповідного стовпця й рядка ставиться позначка.

Крок 4. *Знаходження істотних імплікант.*

Коли якийсь стовпчик складеної на кроці 3 таблиці має тільки одну позначку, то первинна імпліканта у відповідному рядку є істотною, оскільки без неї не буде отримано усієї множини заданих мінтермів. Стовпці, що відповідають істотним імплікантам, з таблиці викреслюються.

Крок 5. *Викреслювання зайвих стовців.*

Якщо після виконання кроку 4 таблиця містить два однакові стовпці (тобто такі, що мають позначки в однакових рядках), то один з них викреслюється. Покриття стовпця, що залишився, буде здійснювати відкинутий мінтерм.

Крок 6. *Викреслювання зайвих первинних імплікант.*

Якщо після відкидання якихось стовпців на кроці 5 з'являються рядки, що не мають жодної позначки, то первинні імпліканти, відповідні цим рядкам, вилучаються з подальшого розгляду, оскільки вони не покривають мінтерми, котрі залишилися в розгляді.

Крок 7. *Вибір мінімального покриття.*

У таблиці, яку було складено на кроці 3 і скореговано на кроках 4, 5 і 6, вибирається така сукупність первинних імплікант, що включає позначки в усіх стовпцях (принаймні по одній позначці в кожному). Якщо існує кілька

можливих варіантів такого вибору, то перевага надається варіанту покриття з мінімальним сумарним числом букв в імплікантах, котрі утворюють покриття.

Таким чином, тупикова форма заданої функції буде складатися із суми істотних імплікант (крок 4) і первинних імплікант які покривають мінтерми, що залишилися (крок 2).

Приклад 9. Знайти тупикову форму для поданої нижче ДНФ.

$f(x_1,x_2,x_3,x_4)=x_2\overline{x_3}+\overline{x_1}x_3x_4+x_1\overline{x_2}x_4+x_1x_2\overline{x_3}x_4+\overline{x_1}x_2\overline{x_3}\,\overline{x_4}$.

Розв'язування

Крок 1

Знаходимо ДДНФ заданої функції $f(x_1,x_2,x_{3,}x_4)$ (див. приклад 8), таким чином:

$$f(x_1,x_2,x_3,x_4)=\left(x_1+\overline{x_1}\right)x_2\overline{x_3}\left(x_4+\overline{x_4}\right)+\overline{x_1}\left(x_2+\overline{x_2}\right)x_3x_4+$$

$$+x_1\overline{x_2}\left(x_3+\overline{x_3}\right)x_4+x_1x_2\overline{x_3}x_4+\overline{x_1}x_2\overline{x_3}\,\overline{x_4}=$$

$$=x_1x_2\overline{x_3}x_4+x_1x_2\overline{x_3}\,\overline{x_4}+\overline{x_1}x_2\overline{x_3}x_4+\overline{x_1}x_2x_3x_4+\overline{x_1}x_2x_3x_4+x_1\overline{x_2}x_3x_4+x_1\overline{x_2}x_3x_4$$

Крок 2

Визначаємо первинні імпліканти. Для цього спочатку складемо таблицю вихідних мінтермів. Вона буде містити стільки рядків і стовпців, скільки мінтермів у ДДНФ заданої функції. На перехресті рядка й стовпця запишемо загальну частину відповідних мінтермів. Наприклад, перший стовпчик відповідає мінтерму $x_1x_2\overline{x_3}x_4$ а другий рядок – мінтерму $x_1x_2\overline{x_3}\,\overline{x_4}$, вони відрізняються один від одного лише змінною x_4, вона входить до першого мінтерму в прямому вигляді, а до другого – в інверсному. Тому в місці перетину першого стовпця й другого рядка ми запишемо тільки спільну частину цих мінтермів, а саме: $x_1x_2\overline{x_3}$. Так заповнюються всі клітинки таблиці.

Вихідні терми	$x_1x_2\overline{x_3}x_4$	$x_1x_2\overline{x_3}\,\overline{x_4}$	$\overline{x_1}x_2\overline{x_3}x_4$	$\overline{x_1}x_2\overline{x_3}\,\overline{x_4}$	$\overline{x_1}x_2x_3x_4$	$\overline{x_1}x_2x_3x_4$	$\overline{x_1}x_2x_3x_4$	$\overline{x_1}x_2x_3x_4$
$x_1x_2\overline{x_3}x_4$		$x_1x_2\overline{x_3}$	$x_2\overline{x_3}x_4$					$x_1\overline{x_3}x_4$
$x_1x_2\overline{x_3}\,\overline{x_4}$	$x_1x_2\overline{x_3}$			$x_2\overline{x_3}\,\overline{x_4}$				
$\overline{x_1}x_2\overline{x_3}x_4$	$x_2\overline{x_3}x_4$			$\overline{x_1}x_2\overline{x_3}$	$\overline{x_1}x_2x_4$			
$\overline{x_1}x_2\overline{x_3}\,\overline{x_4}$		$x_2\overline{x_3}\,\overline{x_4}$	$\overline{x_1}x_2\overline{x_3}$					
$\overline{x_1}x_2x_3x_4$			$\overline{x_1}x_2x_4$			$\overline{x_1}x_3x_4$		
$\overline{x_1}\,\overline{x_2}x_3x_4$					$\overline{x_1}x_3x_4$		$\overline{x_2}x_3x_4$	
$x_1\overline{x_2}x_3x_4$						$x_2\overline{x_3}x_4$		$x_2\overline{x_2}x_4$
$x_1\overline{x_2}\,\overline{x_3}x_4$	$x_1\overline{x_3}x_4$						$x_1\overline{x_2}x_4$	

Тепер складемо таблицю первинних імплікант. Включаємо в неї всі імпліканти, отримані в першій таблиці. У нашому випадку їх дев'ять. Далі заповнюємо таблицю так само, як і попередню. Прямокутниками позначено первинні імпліканти, які не склеюються.

Первинна імпліканта	$x_1x_2\overline{x_3}$	$x_2\overline{x_3}x_4$	$x_2\overline{x_3}\,\overline{x_4}$	$\overline{x_1}x_2\overline{x_3}$	$\boxed{x_1\overline{x_3}x_4}$	$\boxed{\overline{x_1}x_3x_4}$	$\boxed{\overline{x_2}x_3x_4}$	$\boxed{x_1\overline{x_2}x_4}$	$\boxed{\overline{x_1}x_2x_4}$
$x_1x_2\overline{x_3}$				$\boxed{x_2\overline{x_3}}$					
$x_2\overline{x_3}x_4$			$x_2\overline{x_3}$						
$x_2\overline{x_3}\,\overline{x_4}$		$x_2\overline{x_3}$							
$\overline{x_1}x_2\overline{x_3}$	$x_2\overline{x_3}$								
$x_1\overline{x_3}x_4$									
$\overline{x_1}x_3x_4$									
$\overline{x_2}x_3x_4$									
$x_1\overline{x_2}x_4$									
$\overline{x_1}x_2x_4$									

Крок 3. Розміщення позначок.

Складаємо таблицю, рядки якої відповідають первинним імплікантам, отриманим на попередньому кроці, а стовпчики – вихідним термам. Якщо відповідна імпліканта міститься в термі – то на перетині цього рядка й стовпця ставимо позначку V. Наприклад, мінтерм $x_1x_2\overline{x_3}x_4$ містить первинні імпліканти $x_2\overline{x_3}x_4$ та $x_2\overline{x_3}$. Тому на перетині відповідних стовпця (у нашому прикладі він перший) і рядків (перший та останній) ставимо позначку V.

Первинні імпліканти	Вихідні терми							
	$x_1x_2\overline{x_3}x_4$	$x_1x_2\overline{x_3}\,\overline{x_4}$	$\overline{x_1}x_2\overline{x_3}x_4$	$\overline{x_1}x_2\overline{x_3}\,\overline{x_4}$	$\overline{x_1}x_2x_3x_4$	$\overline{x_1}\,\overline{x_2}x_3x_4$	$x_1\overline{x_2}x_3x_4$	$x_1\overline{x_2}x_3\overline{x_4}$
$x_2\overline{x_3}x_4$	V							V
$\boxed{\overline{x_1}x_3x_4}$					V	V		
$\overline{x_2}x_3x_4$						V	V	
$\boxed{x_1\overline{x_2}x_4}$							V	V
$\overline{x_1}x_2x_4$			V		V			
$\boxed{x_2\overline{x_3}}$	V	V	V	V				

↑ ↑ ↑ ↑

Крок 4. Знаходження істотних імплікант. До них належить первинна імпліканта $x_2\overline{x_3}$, оскільки в стовпці $\overline{x_1}x_2\overline{x_3}\overline{x_4}$ наявна тільки одна позначка і вона відповідає рядку $x_2\overline{x_3}$.

З останньої таблиці викреслюємо стовпці, що відповідають істотній імпліканті (їх позначено стрілками). Після цього таблиця набуває такого вигляду:

Первинні імпліканти	Вихідні терми			
	$\overline{x_1}x_2x_3x_4$	$\overline{x_1}\overline{x_2}x_3x_4$	$x_1\overline{x_2}x_3x_4$	$x_1\overline{x_2}\overline{x_3}x_4$
$x_2\overline{x_3}x_4$				V
$\overline{x_1}x_3x_4$	V	V		
$\overline{x_2}x_3x_4$		V	V	
$x_1\overline{x_2}x_4$			V	V
$\overline{x_1}x_2x_4$	V			

Крок 5. Не виконується, оскільки в отриманій після викреслювання таблиці немає стовпців, які мають однакові позначки.

Крок 6. Не здійснюємо, тому що в кожному рядку отриманої таблиці є позначки.

Крок 7. Вибір мінімального покриття.

Мінімальне покриття вихідних термів, що залишилися, здійснюють первинні імпліканти $\overline{x_1}x_3x_4$ й $x_1\overline{x_2}x_4$ (у таблиці їх позначено прямокутниами).

Таким чином, тупикова форма заданої ДНФ включає істотні імпліканти (у нашому випадку $x_2\overline{x_3}$) та імпліканти, які здійснюють мінімальне покриття ($\overline{x_1}x_3x_4$ й $x_1\overline{x_2}x_4$). Вона має такий вигляд:

$$f\left(x_1,x_2,x_3,x_4\right)=x_2\overline{x_3}+\overline{x_1}x_3x_4+x_1\overline{x_2}x_4.$$

Подальша мінімізація отриманої ДНФ можлива, наприклад, шляхом уведення в її запис дужок: $f\left(x_1,x_2,x_3,x_4\right)=x_2\overline{x_3}+\left(\overline{x_1}x_3+x_1\overline{x_2}\right)x_4$.

3.2.2. Метод Квайна – Мак-Класкі

Недоліком методу Квайна є необхідність повного попарного порівняння всіх мінтермів на етапі визначення первинних імплікант. Це стає суттєвим із зростанням числа мінтермів, оскільки збільшується і число попарних порівнянь.

Числова форма функцій алгебри логіки дозволяє спростити процес визначення первинних імплікант. Це враховано в методі Квайна – Мак-Класкі.

Його особливість – формалізація пошуку простих (первинних) імплікант. ЇЇ виконують таким чином:

– усі конституенти одиниці з ДДНФ булевої функції f записують у вигляді їх двійкових номерів;

– усі номери розбиваються на неперетинні групи. Ознака створення i-ї групи: наявність i одиниць у кожному двійковому номері конституенти одиниці;

– склеювання проводять тільки між номерами сусідніх груп;

– склеювані номери позначаються в який-небудь спосіб (наприклад, їх закреслюють);

– склеювання проводять всілякі, як і в методі Квайна. Непозначені після склеювання номери і є простими імплікантами.

Зазначимо, що групу термів, ранг яких дорівнює $n - r$, називають r-кубом.

Опишемо алгоритм методу Квайна – Мак-Класкі.

Крок 1. Усі мінтерми ДДНФ записуються у вигляді їх двійкових номерів, а саме: змінній x ставиться у відповідність 1, а її інверсії $\bar{x}$ – 0. Наприклад, мінтерму $\overline{x_1}x_2x_3x_4$ відповідає двійковий номер 0111.

Крок 2. Пошук первинних імплікант.

Усі двійкові номери розбиваються відповідно до числа одиниць на неперетинні групи. При цьому в i-ту групу ввійдуть усі номери (набори), що мають у своєму двійковому записі i одиниць. Оскільки умовою створення r-куба є наявність розбіжності тільки за однією координатою (тобто в одному двійковому розряді) в $(r-1)$-кубах та однакових незалежних координат, то групи, які відрізняються двома й більше розрядами, просто не варто порівнювати. Отже, попарне порівняння можна робити тільки з наборами сусідніх за номерами груп.

Кроки 3 – 7 виконуються аналогічно тим самим крокам в алгоритмі методу Квайна. Різниця лише в тому, що в них фігурують числові мінтерми.

Приклад 10. Знайти тупикову ДНФ функції, записаної аналітично, а саме:

$$f(x_1,x_2,x_3,x_4)=\bigvee_1(3,4,5,7,9,11,12,13).$$

Тут у дужках перераховано десяткові еквіваленти наборів, на яких функція f дорівнює 1.

Розв'язування

Крок 1. Спочатку випишемо всі 0-куби (терми максимального рангу), а саме:

$$K^0 = \{0011, 0100, 0101, 0111, 1001, 1011, 1100, 1101\}.$$

Розіб'ємо 0-куби на групи за кількістю одиниць у кожному двійковому наборі, у результаті чого утворюємо три групи:

$$K_1^0 = \{0\ \ 1\ \ 0\ \ 0\}; \qquad K_2^0 = \begin{Bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{Bmatrix}; \qquad K_3^0 = \begin{Bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{Bmatrix}.$$

Крок 2. Пошук первинних імплікант.

а) порівнюємо K_1^0 й K_2^0 і знаходимо мінтерми, які відрізняються тільки в одному розряді. Набори, які склеюються, позначимо зірочкою (*).

$$\begin{array}{c} 0\ \ 1\ \ 0\ \ 0* \end{array} \qquad \begin{array}{c} 0\ \ 0\ \ 1\ \ 1 \\ 0\ \ 1\ \ 0\ \ 1* \\ 1\ \ 0\ \ 0\ \ 1 \\ 1\ \ 1\ \ 0\ \ 0* \end{array}$$

За результатами порівняння будуємо 1-куби, у яких поглинута координата замінюється символом «–»:

$$\begin{array}{c} 0\ \ 1\ \ 0\ \ 0 \\ \underline{0\ \ 1\ \ 0\ \ 1} \\ 0\ \ 1\ \ 0\ \ - \end{array} \qquad \begin{array}{c} 0\ \ 1\ \ 0\ \ 0 \\ \underline{1\ \ 1\ \ 0\ \ 0} \\ -\ \ 1\ \ 0\ \ 0 \end{array}$$

б) Тепер порівняємо K_2^0 й K_3^0.

$$\begin{array}{c} 0\ \ 0\ \ 1\ \ 1* \\ 0\ \ 1\ \ 0\ \ 1* \\ 1\ \ 0\ \ 0\ \ 1* \\ 1\ \ 1\ \ 0\ \ 0* \end{array} \qquad \begin{array}{c} 0\ \ 1\ \ 1\ \ 1\ * \\ 1\ \ 0\ \ 1\ \ 1\ * \\ 1\ \ 1\ \ 0\ \ 1\ * \end{array}$$

На підставі порівняння будуємо 1-куби:

$$\begin{array}{c} 0\ \ 0\ \ 1\ \ 1 \\ \underline{0\ \ 1\ \ 1\ \ 1} \\ 0\ \ -\ \ 1\ \ 1 \end{array} \qquad \begin{array}{c} 0\ \ 0\ \ 1\ \ 1 \\ \underline{1\ \ 0\ \ 1\ \ 1} \\ -\ \ 0\ \ 1\ \ 1 \end{array}$$

$$\begin{array}{cccc} 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ \hline 0 & 1 & - & 1 \end{array} \qquad \begin{array}{cccc} 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ \hline - & 1 & 0 & 1 \end{array}$$

$$\begin{array}{cccc} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ \hline 1 & 0 & - & 1 \end{array} \qquad \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ \hline 1 & - & 0 & 1 \end{array}$$

$$\begin{array}{cccc} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ \hline 1 & 1 & 0 & - \end{array}.$$

Оскільки всі мінтерми 4-го рангу були «склеювані», тобто непозначених мінтермів не залишилось, робимо висновок, що первинних імплікант 4-го рангу немає.

Тепер розіб'ємо всі отримані тут 1-куби на групи відповідно до положення незалежної координати «–». У нас вийшло чотири групи:

$$K_1^1 = \begin{Bmatrix} 0 & 1 & 0 & - \\ 1 & 1 & 0 & - \end{Bmatrix},\ K_2^1 = \begin{Bmatrix} 0 & 1 & - & 1 \\ 1 & 0 & - & 1 \end{Bmatrix},\ K_3^1 = \begin{Bmatrix} 0 & - & 1 & 1 \\ 1 & - & 0 & 1 \end{Bmatrix},\ K_4^1 = \begin{Bmatrix} - & 1 & 0 & 0 \\ - & 0 & 1 & 1 \\ - & 1 & 0 & 1 \end{Bmatrix}.$$

Порівняння усередині кожної з груп K_1^1, K_2^1, K_3^1 і K_4^1 дає такий результат:

$$K_1^1: \begin{array}{cccc} 0 & 1 & 0 & - \\ 1 & 1 & 0 & - \\ \hline - & 1 & 0 & - \end{array} \qquad K_2^1: \begin{array}{cccc} 0 & 1 & - & 1 \times \\ 1 & 0 & - & 1 \times \end{array} \qquad K_3^1: \begin{array}{cccc} 0 & - & 1 & 1 \times \\ 1 & - & 0 & 1 \times \end{array}$$

$$K_4^1: \begin{array}{cccc} - & 1 & 0 & 0 \\ - & 1 & 0 & 1 \\ \hline - & 1 & 0 & - \end{array} \qquad - \ 0 \ 1 \ 1 \times .$$

Символом « × » позначено мінтерми, які не можна склеїти, отже, вони являють собою первинні імпліканти 3-го рангу.

За результатами склеювання отримано один 2-куб $\{- \ 1 \ 0 \ -\}$.

Таким чином, первинними імплікантами будуть:

1-куби: $0\ 1 - 1,\ \ 1\ 0 - 1,\ \ 0 - 1\ 1,\ \ 1 - 0\ 1,\ \ - 0\ 1\ 1$

і
2-куб: – 1 0 – .

Крок 3. Розміщення позначок у таблиці.
Цей крок здійснюється так само, як і в алгоритмі Квайна (див. приклад 9).

Первинні імплікан-ти	Вихідні терми							
	0 1 0 0	0 1 0 1	1 1 0 0	1 1 0 1	0 0 1 1	0 1 1 1	1 0 0 1	1 0 1 1
0 1 – 1		V				V		
1 0 – 1							V	V
0 – 1 1					V	V		
1 – 0 1				V			V	
– 0 1 1					V			V
–1 0 –	V	V	V	V				

Крок 4. Істотною імплікантою є 2-куб: $\{-\ 1\ 0\ -\}$, що відповідає кон'юнкції $x_2\overline{x_3}$, оскільки стовпці, вихідних термів 0100 і 1100 містять позначки тільки в рядку, відповідному цій імпліканті.

Викреслюємо стовпці, які містять позначки в рядку імпліканти {–1 0 –}. У таблиці їх виділено іншим кольором.

Кроки 5 і 6 не потрібні, враховуючи вигляд отриманої таблиці.

Крок 7. Вибір мінімального покриття термів, що залишилися.

Вочевидь, мінімальне покриття термів, які залишились, здійснюють імпліканти {10–1} і {0–11}. Їм відповідають кон'юнкції: $x_1\overline{x_2}x_4$ і $\overline{x_1}x_3x_4$.

Отже, тупикова форма вихідної функції має такий вигляд:

$$f\left(x_1,x_2,x_3,x_4\right)=x_2\overline{x_3}+x_1\overline{x_2}x_4+\overline{x_1}x_3x_4.$$

3.2.3. Метод Вейча – Карно

Цей метод передбачає побудову тупикової форми логічної функції за допомогою спеціальної таблиці, яка називається *діаграмою* (*картою*) *Вейча – Карно* і являє собою спеціально перебудовану таблицю істинності функції. Ця діаграма виступає графічним способом мінімізації функції і забезпечує відносну простоту роботи із виразами великого обсягу.

Опишемо алгоритм мінімізації логічних функцій за допомогою діаграми Вейча – Карно.

Крок 1. Будують прямокутну систему координат, вісь ординат якої збігається з лівим вертикальним краєм діаграми, а вісь абсцис – з нижнім краєм.

Крок 2. У даній системі координат зображують прямокутну таблицю, рядки й стовпчики якої нумеруються двійковими числами. Кожному двійковому розряду ставиться у відповідність певна логічна змінна.

За таких умов вісь ординат є віссю складного аргументу, який включає (коли n парне) $n/2$ перших логічних змінних: $x_1, x_2, ..., x_{n/2}$, а вісь абсцис служить віссю зміни складного аргументу, що містить $n/2$ наступних логічних змінних, а саме: $x_{\frac{n}{2}+1}, x_{\frac{n}{2}+2}, ..., x_n$. При непарному значенні n складні аргументи включають неоднакову кількість змінних.

Для того, щоб за допомогою описаної вище таблиці можна було не тільки записати будь-яку функцію, але й зробити її мінімізацію, необхідно виконати одну дуже важливу *умову спряженого сусідства* – за якою у сусідніх по вертикалі або по горизонталі клітинках (у фізичному, звичайно, сенсі) повинні перебувати сусідні конституенти (у логіко-математичному сенсі)

Крок 3. З метою виконання умови спряженого сусідства замінюємо двійковий код, яким були позначені стовпці й рядки діаграми, двійковим циклічним *кодом Грея*, користуючись при цьому записаним нижче правилом.

Цей двійковий код має ту особливість, що сусідні два числа відрізняються одне від одного тільки в одному розряді.

Правило перетворення двійкового коду в код Грея

1. Найстарша значуща цифра числа в коді Грея збігається із найстаршою значущою цифрою цього самого числа у двійковому коді.
2. Цифра в будь-якому іншому, молодшому розряді числа коду Грея має такі ознаки:

– збігається з відповідною цифрою у двійковому коді, якщо ліворуч від даної цифри коду Грея розташовано парне число одиниць;

– збігається із запереченням відповідної цифри у двійковому коді, якщо ліворуч від даної цифри коду Грея міститься непарна кількість одиниць;

Наприклад, послідовність у двійковому коді становить: 10110, а перетворення її на код Грея: 11101.

Застосування коду Грея до нумерації клітинок по осях координат у діаграмі Вейча – Карно забезпечує розміщення в сусідніх клітинках діаграми сусідніх членів ДД(К)НФ будь-якої логічної функції.

Крок 4. У самих клітинках діаграми необхідно проставити значення істинності реалізацій функції на кожному з наборів аргументів. Причому, якщо функцію задано за допомогою ДДНФ, то досить проставити тільки одиниці, а коли функцію задано у ДКНФ, тільки нулі.

Крок 5. Проводиться склеювання мінтермів таким чином, що 2^r сусідніх клітинок (0-кубів), об'єднуючись, утворюють r-куб (r = 1, 2, …).

Слід зауважити, що коли кількість змінних більша або дорівнює п'яти, відобразити графічно функцію у вигляді єдиної плоскої діаграми (карти) неможливо. У таких випадках будують комбіновану діаграму, яка складається із сукупності більш простих діаграм, наприклад, чотиривимірних. Тоді процедура мінімізації буде полягати в тому, що спочатку знаходять мінімальні форми всередині чотиривимірних кубів, а потім, розширюючи поняття сусідніх клітинок, відшукують терми мінімального рангу для сукупності діаграм.

Приклад 11. Знайти тупикову ДНФ функції, записаної в аналітичному вигляді:

$$f(x_1, x_2, x_3, x_4) = \bigvee_1 (0,1,4,5,7,10,11,13,15).$$

Розв'язування

Застосуємо діаграми Вейча – Карно.

Кроки 1, 2.

Складемо таблицю, перенумерувавши її рядки й стовбці відповідно до коду Грея (див. рис. 3.1).

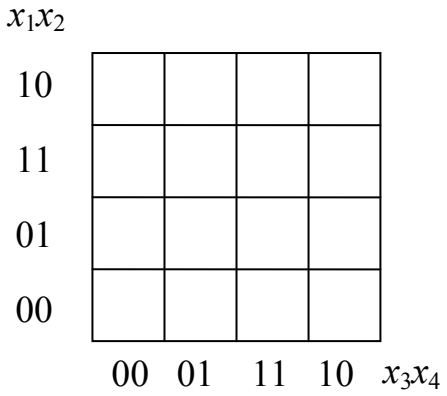


Рис. 3.1. Мінімізація логічних функцій методом Вейча – Карно (кроки 1, 2)

Кроки 3, 4, 5.

Даній функції відповідають такі 0-куби:

$K^0 = \{0000, 0001, 0100, 0101, 0111, 1010, 1011, 1101, 1111\}$. Позначимо їх на діаграмі, записуючи у відповідні клітинки одиниці (рис. 3.2), і мінімізуємо функцію відповідно до сформульованих вище правил.

Для цього виконуємо склеювання суміжних наборів, у клітинках яких записано одиниці. Зони склеювання позначено в таблиці (див. рис. 3.2) прямокутниками. При склеюванні діємо таким чином: спочатку беремо одну з позначених зон і дивимось, які змінні залишаються сталими в її межах, виписуємо їх кон'юнкцію, причому якщо змінна в даній зоні нульова, то беремо

її інверсію. Потім розглядаємо наступну зону і т. д. Кон'юнкції зон об'єднуємо за допомогою диз'юнкції.

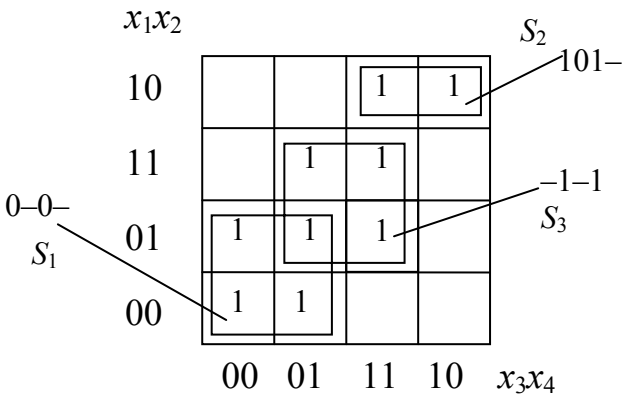


Рис. 3.2. Мінімізація логічних функцій методом Вейча – Карно (кроки 3, 4, 5)

У нашому прикладі в зоні S_1 незмінними залишаються x_1 та x_3. Відповідна їй кон'юнкція має вигляд: $\overline{x}_1\overline{x}_3$. Аналогічно для зони S_2 отримуємо кон'юнкцію $x_1\overline{x}_2x_3$, а для S_3 – x_2x_4.

Таким чином, унаслідок склеювання отримуємо таку тупикову форму вихідної функції:

$$f\left(x_1,x_2,x_3,x_4\right)=\overline{x}_1\overline{x}_3+x_2x_4+x_1\overline{x}_2x_3.$$

Зауважимо, що при визначенні двоклітинного куба необхідно розрізняти, чи він горизонтальний, чи вертикальний. Перший випадок (горизонтальний куб) відповідає тому, що сталі значення мають змінні рядка (у нашому прикладі це куб S_2). Другий випадок (вертикальний куб) відповідає тому, що сталі значення мають змінні стовпця.

§ 3.3. Мінімізація частково визначених двійкових функцій

У практичній діяльності нерідко доводиться мати справи з функціями, які визначено не для всіх наборів змінних. Подібні випадки трапляються тоді, коли за умовами функціонування деякі з наборів не використовуються і тому байдуже, якого значення на них набуває функція. Цю обставину можна використовувати при мінімізації функції, задавши її на «байдужих» наборах таким чином, щоб забезпечити найбільш економічну реалізацію.

Пояснимо сенс такого прийому. Нехай дано частково визначену функцію: $f=f\left(x_1,x_2,...,x_n\right)$. Позначимо через φ_1: $\varphi_1=\varphi_1\left(x_1,x_2,...,x_n\right)$, функцію, яку задано на всіх «байдужих» наборах одиницями; а через φ_0: $\varphi_0=\varphi_0\left(x_1,x_2,...,x_n\right)$, функцію, яку задано на всіх «байдужих» наборах нулями. Задача оптимального

визначення даної функції f зводиться до вибору із скороченого покриття для функції φ_1 мінімальної кількості кубів максимальної розмірності, сукупність яких покривала б усі вершини функції φ_0.

Ця сукупність якраз і утворює мінімальне покриття частково визначеної функції f. При цьому воно може покривати й деякі 0-куби, відповідні «байдужим» наборам, що свідчить про те, що функцію задано на цих наборах одиничними значеннями.

Для мінімізації частково визначених функцій використовують розглянуті раніше методи мінімізації Квайна, Квайна – Мак-Класкі й діаграми Вейча – Карно.

Приклад 12. Знайти тупикову форму функції чотирьох змінних, яку задано таким чином:

$$f(x_1, x_2, x_3, x_4) = \bigvee_1 \left(0, 1^*, 2^*, 4^*, 6, 7^*, 8^*, 9, 11^*, 13^*, 14, 15^*\right).$$

У цьому виразі десяткові номери наборів, на яких функцію недовизначено, записано зі знаком «$*$».

Розв'язування

Складемо таблицю істинності заданої функції f, включаючи значення функцій φ_1 і φ_0.

№	x_1	x_2	x_3	x_4	f	φ_1	φ_0
0	0	0	0	0	1	1	1
1	0	0	0	1	*	1	0
2	0	0	1	0	*	1	0
3	0	0	1	1	0	0	0
4	0	1	0	0	*	1	0
5	0	1	0	1	0	0	0
6	0	1	1	0	1	1	1
7	0	1	1	1	*	1	0

№	x_1	x_2	x_3	x_4	f	φ_1	φ_0
8	1	0	0	0	*	1	0
9	1	0	0	1	1	1	1
10	1	0	1	0	0	0	0
11	1	0	1	1	*	1	0
12	1	1	0	0	0	0	0
13	1	1	0	1	*	1	0
14	1	1	1	0	1	1	1
15	1	1	1	1	*	1	0

Крок 1. Об'єднаємо 0-куби в групи за кількістю одиниць у кожному двійковому наборі, тобто

$$K_0^0 = \{0 \quad 0 \quad 0 \quad 0\}; \qquad K_1^0 = \begin{Bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{Bmatrix}; \qquad K_2^0 = \begin{Bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{Bmatrix};$$

$$K_3^0 = \begin{Bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{Bmatrix}; \quad K_4^0 = \{1 \quad 1 \quad 1 \quad 1\},$$

Крок 2. Пошук первинних імплікант, а саме:

а) Порівнюємо K_0^0 й K_1^0 (набори, які склеюються, відзначимо символом $*$);

$$0 \quad 0 \quad 0 \quad 0 \; * \qquad \begin{matrix} 0 & 1 & 0 & 0 & * \\ 0 & 0 & 0 & 1 & * \\ 0 & 0 & 1 & 0 & * \\ 1 & 0 & 0 & 0 & * \end{matrix}$$

На підставі порівняння будуємо 1-куби, у яких поглинуту координату замінюємо символом «–», а саме:

0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
0 1 0 0	0 0 0 1	0 0 1 0	1 0 0 0
0 – 0 0	0 0 0 –	0 0 – 0	– 0 0 0

б) Порівнюємо K_1^0 й K_2^0:

$$\begin{matrix} 0 & 0 & 0 & 1 & * \\ 0 & 0 & 1 & 0 & * \\ 0 & 1 & 0 & 0 & * \\ 1 & 0 & 0 & 0 & * \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 0 & * \\ 1 & 0 & 0 & 1 & * \end{matrix}$$

На базі порівняння будуємо такі 1-куби:

0 0 1 0	0 1 0 0	0 0 0 1	1 0 0 0
0 1 1 0	0 1 1 0	1 0 0 1	1 0 0 1
0 – 1 0	0 1 – 0	– 0 0 1	1 0 0 –

в) Порівнюємо K_2^0 й K_3^0:

0	1	1	0	*		0	1	1	1 *
1	0	0	1	*		1	0	1	1 *
						1	1	0	1 *
						1	1	1	0 *

На підставі порівняння будуємо 1-куби:

0	1	1	0		0	1	1	0		1	0	0	1		1	0	0	1
0	1	1	1		1	1	1	0		1	0	1	1		1	1	0	1
0	1	1	–		–	1	1	0		1	0	–	1		1	–	0	1

г) Порівнюємо K_3^0 й K_4^0:

0	1	1	1	*					
1	0	1	1	*					
1	1	0	1	*		1	1	1	1 *
1	1	1	0	*					

На основі порівняння будуємо 1-куби:

0	1	1	1		1	0	1	1		1	1	0	1		1	1	1	0
1	1	1	1		1	1	1	1		1	1	1	1		1	1	1	1
–	1	1	1		1	–	1	1		1	1	–	1		1	1	1	–

Як бачимо, первинних імплікант четвертого рангу функції φ_1 немає.

Розіб'ємо всі 1-куби на групи відповідно до числа одиниць, враховуючи положення незалежної змінної «–», а саме:

$$K_1^1 = \begin{Bmatrix} 0 & 0 & 0 & – \\ 1 & 0 & 0 & – \\ 0 & 1 & 1 & – \\ 1 & 1 & 1 & – \end{Bmatrix}, \quad K_2^1 = \begin{Bmatrix} 0 & 0 & – & 0 \\ 0 & 1 & – & 0 \\ 1 & 0 & – & 1 \\ 1 & 1 & – & 1 \end{Bmatrix},$$

$$K_3^1 = \begin{Bmatrix} 0 & – & 0 & 0 \\ 0 & – & 1 & 0 \\ 1 & – & 0 & 1 \\ 1 & – & 1 & 1 \end{Bmatrix}, \quad K_4^1 = \begin{Bmatrix} – & 0 & 0 & 0 \\ – & 0 & 0 & 1 \\ – & 1 & 1 & 0 \\ – & 1 & 1 & 1 \end{Bmatrix}.$$

Порівняння груп, що мають різну кількість одиниць, усередині кубів K_1^1, K_2^1, K_3^1 і K_4^1 зумовлює такий результат:

0 0 0 –	0 1 1 –	0 0 – 0	1 0 – 1	0 – 0 0
1 0 0 –	1 1 1 –	0 1 – 0	1 1 – 1	1 – 1 0
– 0 0 –	– 1 1 –	0 – – 0	1 – – 1	0 – – 0

1 – 0 1	– 0 0 0	– 1 1 1
1 – 1 1	– 0 0 1	– 1 1 0
1 – – 1	– 0 0 –	– 1 1 –

Отже, первинних імплікант функції φ_1 третього рангу немає.

Отже, за результатами склеювання отримано такі первинні імпліканти:

$$\{-0\,0-\},\ \{0--0\},\ \{-1\,1-\},\ \{1--1\}.$$

Крок 3. Розміщення позначок.

Результати виконання цього кроку показано в поданій нижче таблиці.

Первинні імпліканти φ_1	Вихідні терми φ_0			
	0 1 1 0	1 1 1 0	0 0 0 0	1 0 0 1
– 0 0 –			V	V
– 1 1 –	V	V		
1 – – 1				V
0 – – 0	V		V	

Крок 4. Пошук істотних імплікант.

Істотною тут виявляється первинна імпліканта { – 1 1 – }. Із таблиці викреслюємо відповідні їй стовпці.

Кроки 5 і 6 не потрібні.

Крок 7. Вибір мінімального покриття.

Мінімальне покриття термів, що залишилися, здійснює первинна імпліканта { – 0 0 – }.

Таким чином, тупикова форма заданої частково визначеної функції має такий вигляд:

$$f(x_1, x_2, x_3, x_4) = x_2x_3 + \bar{x}_2\bar{x}_3.$$

Тепер розв'яжемо цю задачу, використовуючи діаграми Вейча – Карно.

Розв'язування

Побудуємо діаграми Вейча – Карно функцій φ_1 та φ_0.

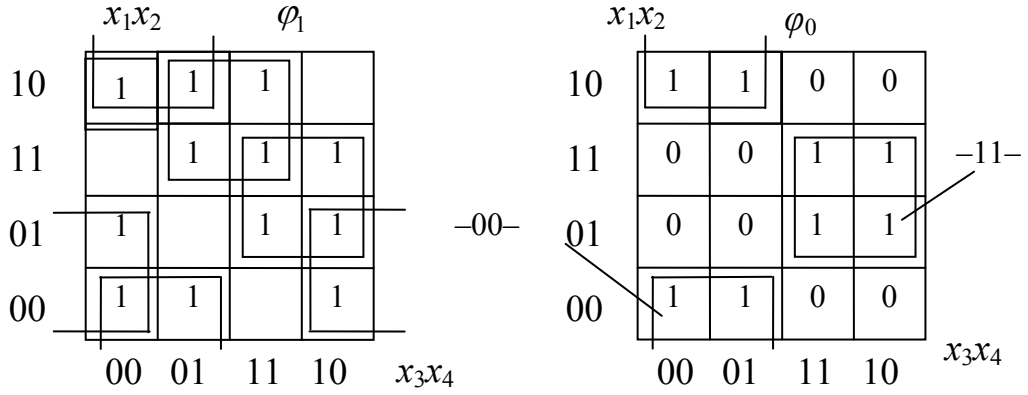


Рис. 3.3. Діаграми Вейча – Карно функцій φ_1 та φ_0

Діаграма Вейча-Карно, що характеризує функцію φ_0, містить результати операції пошуку вихідної частково визначеної функції f, які забезпечують одержання тупикової ДНФ такого вигляду: $f(x_1, x_2, x_3, x_4) = x_2x_3 + \bar{x}_2\bar{x}_3$.

Отриманий тут результат збігається з результатом мінімізації, отриманим за допомогою методу Мак-Класкі.

§ 3.4. Пошук мінімальних КНФ

У деяких випадках схема, що реалізує мінімальну КНФ, виявляється більш економічною, ніж схема, що реалізує ДНФ тієї самої логічної функції. Тому з метою одержання оптимального розв'язку бажано знайти мінімальні ДНФ і КНФ логічної функції.

Описані вище методи мінімізації ДДНФ можуть бути застосовані й для мінімізації ДКНФ, якщо користуватися двоїстими формами відповідних логічних законів, а замість конституент одиниці використовувати конституенти нуля.

Разом з тим на практиці дуже часто замість використання методів мінімізації КНФ застосовують підхід, що дозволяє виявляти тупикові КНФ за допомогою методів мінімізації ДДНФ і правила де Моргана.

Цей підхід характеризується трьома кроками, а саме:

Крок 1. Записуємо заперечення заданої функції як диз'юнкцію елементарних кон'юнкцій максимального рангу, що мають номери тих наборів, на яких логічна функція набуває нульового значення.

Крок 2. Отриманий вираз мінімізуємо за допомогою кожного з методів мінімізації ДДНФ. Внаслідок цього отримуємо мінімальні диз'юнктивні форми заперечення заданої функції.

Крок 3. Від мінімальних (тупикових) ДНФ заперечення заданої функції за допомогою правила де Моргана $\left(\overline{x}_1\overline{x}_2 = \overline{x}_1 + \overline{x}_2, \overline{x_1 + x_2} = \overline{x}_1\overline{x}_2\right)$ переходимо до тупикових КНФ заданої функції.

Приклад 13. Для записаної аналітично ДНФ знайти тупикову КНФ.

$$f\left(x_1, x_2, x_3, x_4\right) = \bigvee_1 (3,4,5,7,9,11,12,13).$$

Розв'язування

З умови задачі випливає, що функція f має нульове значення на таких наборах: 0, 1, 2, 6, 8, 10, 14, 15. Двійкові еквіваленти цих номерів мають такий вигляд: 0000, 0001, 0010, 0110, 1000, 1010, 1110, 1111.

Крок 1. Записуємо заперечення заданої функції:

$$f\left(x_1, x_2, x_3, x_4\right) = \overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4 + \overline{x}_1\overline{x}_2\overline{x}_3 x_4 + \overline{x}_1\overline{x}_2 x_3\overline{x}_4 + \overline{x}_1 x_2\overline{x}_3\overline{x}_4 + x_1\overline{x}_2\overline{x}_3\overline{x}_4 + x_1\overline{x}_2 x_3\overline{x}_4 +$$
$$+ x_1 x_2 x_3\overline{x}_4 + x_1 x_2 x_3 x_4.$$

Крок 2. Знайдемо тупикову ДНФ функції f за допомогою діаграми Вейча – Карно.

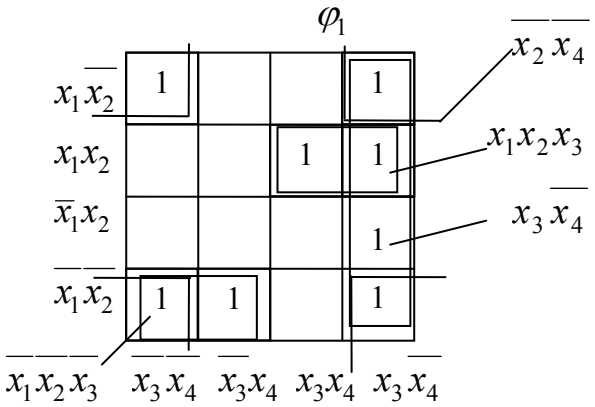


Рис. 3.4. Діаграма Вейча – Карно до прикладу 13

Після мінімізації функція набуває такого вигляду:

$$f\left(x_1x_2x_3x_4\right) = \overline{x}_2\overline{x}_4 + x_3\overline{x}_4 + x_1x_2x_3 + \overline{x}_1\overline{x}_2\overline{x}_3.$$

Крок 3. Використовуючи правило де Моргана, визначимо тупикову КНФ, а саме:

$$f(x_1x_2x_3x_4) = \overline{\overline{x_2}\,\overline{x_4} + x_3\overline{x_4} + x_1x_2x_3 + \overline{x_1}\,\overline{x_2}\,\overline{x_3}} = \overline{\overline{x_2}\,\overline{x_4}} \cdot \overline{x_3\overline{x_4}} \cdot \overline{x_1x_2x_3} \cdot \overline{\overline{x_1}\,\overline{x_2}\,\overline{x_3}} =$$
$$= (x_2 + x_4)(\overline{x_3} + x_4)(\overline{x_1} + \overline{x_2} + \overline{x_3})(x_1 + x_2 + x_3).$$

§ 3.5. Синтез логічних (комбінаційних) схем

Обладнання, яке реалізує елементарні булеві функції, називають *логічними елементами.* Їхні входи відповідають булевим змінним, а вихід – реалізованій функції. Види логічних елементів показано на рис. 3.3.

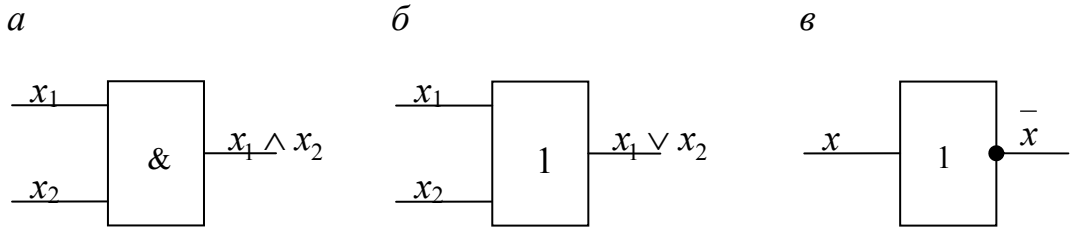


Рис. 3.5. Види логічних елементів: *а* – кон'юнктор, *б* – диз'юнктор, *в* – інвертор

Подібно до суперпозиції функцій, логічні схеми утворюються шляхом суперпозиції елементів за допомогою визначення їх зовнішніх вузлів (полюсів). Коректно побудовані логічні схеми повинні задовольняти такі умови:

– не допускати замкнених контурів, які можуть призвести до неоднозначності на виходах елементів;

– будь-який вхід елемента має бути з'єднаним тільки з одним входом схеми або виходом іншого елемента;

– виходи елементів, що не являють собою виходи схеми та не з'єднані із входами інших елементів, вважаються зайвими й вилучаються зі схеми.

Зауважимо, що записати булеву функцію для заданої логічної схеми досить легко. Так само просто будується й логічна схема даного аналітичного виразу булевої функції. Успішне проектування логічних схем полягає в тому, щоб забезпечити *найбільш економічну* реалізацію булевої функції.

Опишемо алгоритм синтезу логічних схем.

Крок 1. Складаємо таблиці істинності синтезованого вузла згідно з його визначенням, призначенням і словесним описом принципу роботи.

Крок 2. Будуємо математичну формулу логічної функції, що описує роботу синтезованої схеми (вузла) згідно з наявною таблицею істинності.

Крок 3. Аналізуємо отриману функцію з метою побудови різних варіантів її математичного запису (на підставі законів булевої алгебри) та знаходимо найкращий з них відповідно до заданого критерію.

Крок 4. Складаємо функціональну (логічну) схему з елементів «І», «АБО», «НІ».

Алгоритм описано.

Побудова логічної схеми базується на прямому заміщенні елементарних добутків, сум і заперечень кон'юнкторами, диз'юнкторами й інверторами відповідно.

Для отримання аналітичного виразу заданої таблично (у досконалій диз'юнктивній нормальній формі) функції потрібно скласти суму конституент одиниці тих наборів значень вхідних двійкових змінних, для яких реалізації логічної функції f дорівнюють 1, причому символ будь-якої змінної в конституенті береться зі знаком заперечення, якщо конкретне значення змінної x_i у розглянутому наборі – 0.

Для того, щоб записати аналітично функцію, задану таблично у досконалій кон'юнктивній нормальній формі, потрібно скласти логічний добуток конституент нуля тих наборів значень вхідних двійкових змінних, для яких реалізація функції f дорівнює 0, причому символ будь-якої змінної в конституенті береться зі знаком заперечення, якщо конкретне значення змінної x_i в розглянутому наборі дорівнює 1.

Приклад 14. Роботу трьох верстатів координують таким чином, що коли будь-які два з них вийдуть із ладу, то автоматично включається в роботу четвертий (аварійний) верстат. Потрібно синтезувати логічну схему обладнання, яка керує включенням у роботу четвертого верстата.

Розв'язування

Крок 1. Ведемо позначення. Нехай змінні x_1, x_2, x_3 відображають стани верстатів 1, 2, 3 відповідно. Коли змінна x_i дорівнює 1, то це означає, що *i*-й верстат працює справно, а коли нулю – то верстат вийшов з ладу. Складемо таблицю істинності функції f. Одиничне значення функції відповідає включенню в роботу четвертого верстата.

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
f	1	1	1	0	1	0	0	0

Крок 2. З урахуванням даних складеної вище таблиці істинності запишемо ДДНФ і ДКНФ логічної функції, а саме:

$$f(x_1, x_2, x_3) = \overline{x_1}\,\overline{x_2}\,\overline{x_3} + \overline{x_1}\,\overline{x_2}\,x_3 + \overline{x_1}\,x_2\,\overline{x_3} + x_1\,\overline{x_2}\,\overline{x_3},$$

$$f(x_1, x_2, x_3) = \left(x_1 + \overline{x_2} + \overline{x_3}\right)\left(\overline{x_1} + x_2 + \overline{x_3}\right)\left(\overline{x_1} + \overline{x_2} + x_3\right)\left(\overline{x_1} + \overline{x_2} + \overline{x_3}\right).$$

Крок 3. Зробимо мінімізацію ДДНФ і ДКНФ, використовуючи діаграми Вейча – Карно (див. рис. 3.6).

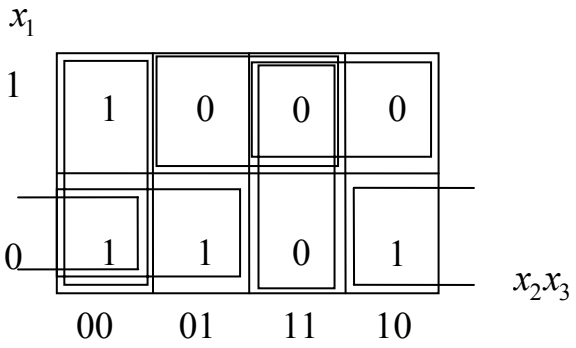


Рис. 3.6. Діаграма Вейча – Карно для функціїї з прикладу 14

Одержимо таку мінімальну ДНФ функції:

$$f\left(x_1,x_2,x_3\right)=\overline{x_1}\,\overline{x_2}+\overline{x_1}\,\overline{x_3}+\overline{x_2}\,\overline{x_3}\ .$$

Беручи заперечення від виразу: $f\left(x_1,x_2,x_3\right)=x_1x_2+x_1x_3+x_2x_3$, й застосувавши закони де Моргана, визначимо мінімальну КНФ в такому вигляді:

$$f\left(x_1x_2x_3\right)=\left(\overline{x_1}+\overline{x_2}\right)\left(\overline{x_1}+\overline{x_3}\right)\left(\overline{x_2}+\overline{x_3}\right).$$

Крок 4. Використовуючи мінімальні ДНФ і КНФ функції, побудуємо відповідні їм логічні схеми. Процес синтезу виконується у напрямку від вхідних значень змінних до отримання вихідних значень. Спочатку використовуючи вхідні змінні, отримують необхідні інверсні значення, а потім, поетапно застосовуючи необхідні елементи для реалізації операцій, маємо кінцеве значення функції. Отримані схеми подано на рис. 3.7 та 3.8.

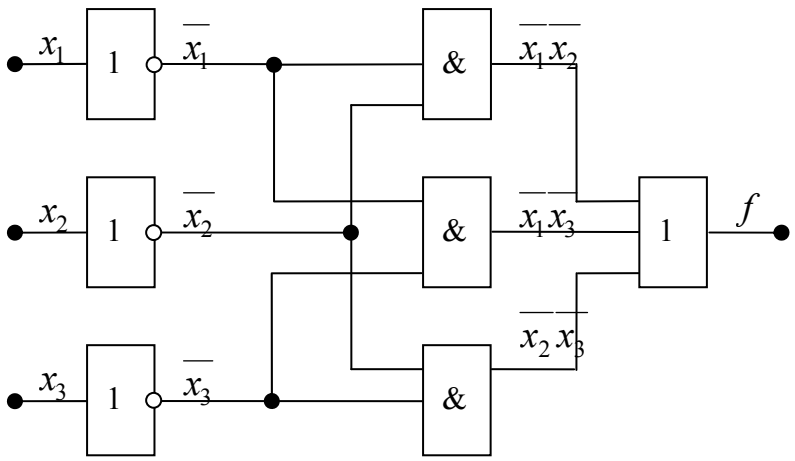


Рис. 3.7. Логічна схема, що базується на мінімальній ДНФ функції (приклад 14)

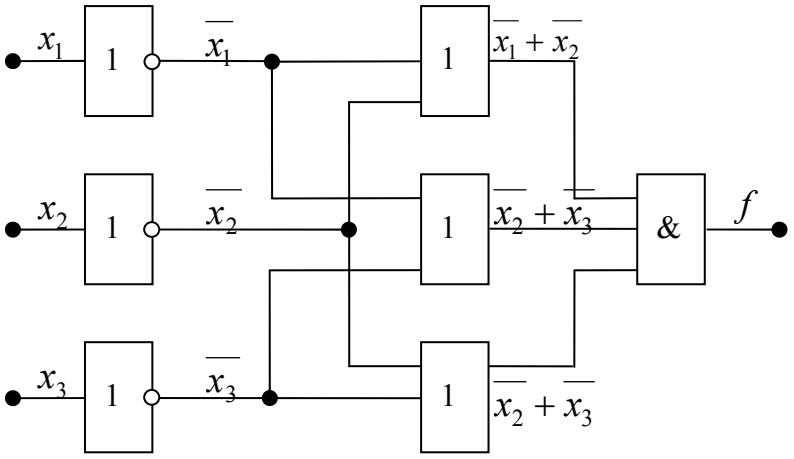


Рис. 3.8. Логічна схема, виконана на базі мінімальної КНФ (приклад 14)

§ 3.6. Синтез скінченних автоматів

Скінченний автомат являє собою математичну модель пристрою із скінченною памяттю. Він переробляє вхідну множину впливів X у множину вихідних сигналів Y, залежно від свого внутрішнього стану S, і характеризується тим, що множини X, Y та S, що містять усі можливі значення входів, виходів та внутрішніх станів, є скінченими. Іншими словами, існує *скінчений перелік* з n можливих значень сигналів на вході, m можливих значень сигналів на виході, а також k можливих внутрішніх станів системи.

Скінченний автомат дуже легко задати у вигляді таблиці розміру $n \times k$, у кожній клітинці якої в чисельнику міститься наступний внутрішній стан системи, а в знаменнику – сигнал на виході (див. рис. 3.9, *а*). Будь-який скінчений автомат можна також подати у вигляді орієнтованого графа, вершинами якого будуть внутрішні стани, а дугами – пари із вхідного і вихідного сигналів (див. рис. 3.9, *б*).

З викладеного раніше випливає, що автомати повинні містити елементи пам’яті, аби зберігати значення стану системи від такту до такту, один чи декілька вхідних та вихідних каналів.

а

$x(n)$ / $s(n)$	x_1	x_2	x_3
S_1	S_2/y_1	S_2/y_2	S_3/y_2
S_2	S_3/y_2	S_2/y_1	S_1/y_2
S_3	S_1/y_2	S_2/y_1	S_3/y_1

б

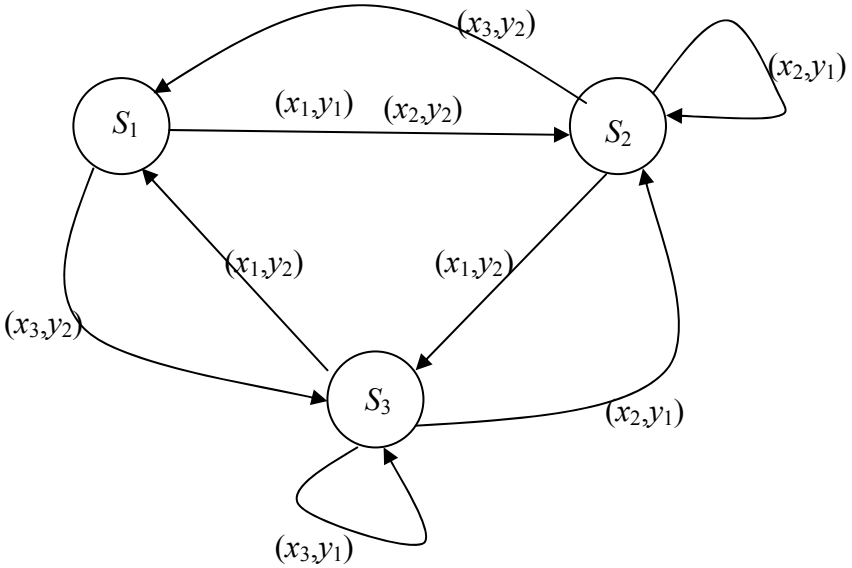


Рис. 3.9. Приклад задання скінченного автомата:
а – за допомогою таблиці переходів; *б* – за допомогою графа

Реалізація скінченних автоматів зводиться до синтезу відповідної комбінаційної схеми, що перетворює вхідні змінні $x(n)$ й $s(n)$ на вихідні змінні $s(n+1)$ й $y(n)$ із заданими характеристичними функціями, тобто

$$s(n+1) = f\left(s(n), x(n)\right),$$

$$y(n) = \varphi\left(s(n), x(n)\right).$$

Щоб зберегти стан $s(n+1)$ до наступного такту, у ланцюг зворотного зв'язку потрібно ввести необхідну кількість елементів пам'яті.

При реалізації автоматів у двійковому структурному алфавіті можна використовувати розглянуті вище методи синтезу комбінаційних схем. Для цього необхідно закодувати кожен стан схеми й записати характеристичні функції у вигляді булевих функцій двійкових змінних. Таке кодування можна здійснити шляхом перетворення загальної таблиці переходу автомата до таблиці істинності в двійковому структурному алфавіті. Якщо елементи множин X, Y, S пронумеровано порядковими числами, починаючи з нуля, то їм відповідають коди, котрі являють собою двійкові еквіваленти цих чисел.

Приклад 15. Синтезувати скінченний автомат, заданий загальною таблицею переходів, яку подано нижче.

$x(n)$ / $s(n)$	0	1	2	3
0	3/0	2/0	1/1	3/0
1	3/0	2/0	1/1	3/0
2	3/0	2/0	2/0	3/0
3	3/0	0/1	0/1	1/1

Розв'язування

Запишемо вихідну таблицю переходів скінченного автомата в такому вигляді:

$x(n)$	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
$s(n)$	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
$s(n+1)$	3	3	3	3	2	2	2	0	1	1	2	0	3	3	3	1
$y(n)$	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1

Замінюючи десяткові числа їх двійковими еквівалентами, одержуємо таблицю істинності (табл. 3.4), у якій $x(n), s(n)$, $s(n+1)$ і $y(n)$ записано двійковими кодами.

Таблиця 3.4

$x(n)$		$s(n)$		$s(n+1)$		$y(n)$
$x_1(n)$	$x_2(n)$	$s_1(n)$	$s_2(n)$	$s_1(n+1)$	$s_2(n+1)$	
0	0	0	0	1	1	0
0	0	0	1	1	1	0
0	0	1	0	1	1	0
0	0	1	1	1	1	0
0	1	0	0	1	0	0
0	1	0	1	1	0	0
0	1	1	0	1	0	0
0	1	1	1	0	0	1
1	0	0	0	0	1	1
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	0	0	1
1	1	0	0	1	1	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	0	1	1

Із цієї таблиці видно, що комбінаційна схема скінченного автомата повинна мати чотири входи, відповідні вхідним змінним $x_1(n)$, $x_2(n)$ і змінним стану $s_1(n)$, $s_2(n)$, а також три виходи, які відповідають змінним стану $s_1(n+1)$, $s_2(n+1)$ і виходу $y(n)$.

Користуючись розглянутим у § 3.5 алгоритмом, синтезуємо комбінаційну схему скінченного автомата. Для цього проведемо роздільну мінімізацію функцій $s_1(n+1)$, $s_2(n+1)$ і $y(n)$ за допомогою діаграм Вейча – Карно (див. рис. 3.10).

Наслідком мінімізації буде такий вигляд функцій:

$$s_1(n+1) = x_2(n)\cdot\overline{s_1}(n) + \overline{x_1}(n)\overline{x_2}(n) + s_1(n)\cdot\overline{s_2}(n);$$

$$s_2(n+1) = \overline{x_1}(n)\cdot\overline{x_2}(n) + x_1(n)x_2(n) + x_1(n)\cdot\overline{s_1}(n);$$

$$y(n) = \overline{s_1}(n+1).$$

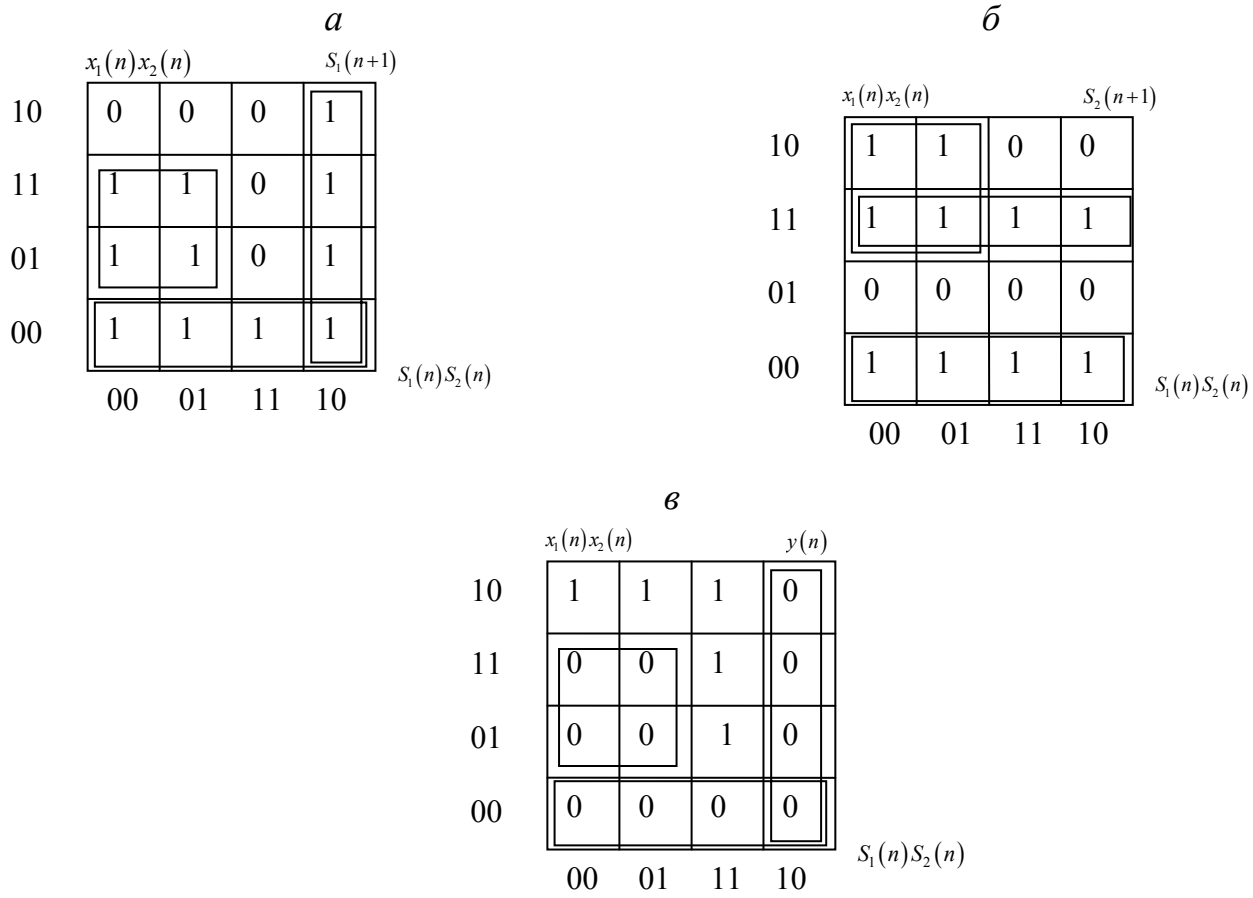


Рис. 3.10. Мінімізація функцій за допомогою діаграм Вейча – Карно:
$а - s_1(n+1)$; $б - s_2(n+1)$; $в - y(n)$

Синтезувавши комбінаційну схему, відповідну вихідній таблиці, а також увівши два елементи затримки C_1 і C_2, одержимо структурну схему скінченного автомата (див. рис. 3.11).

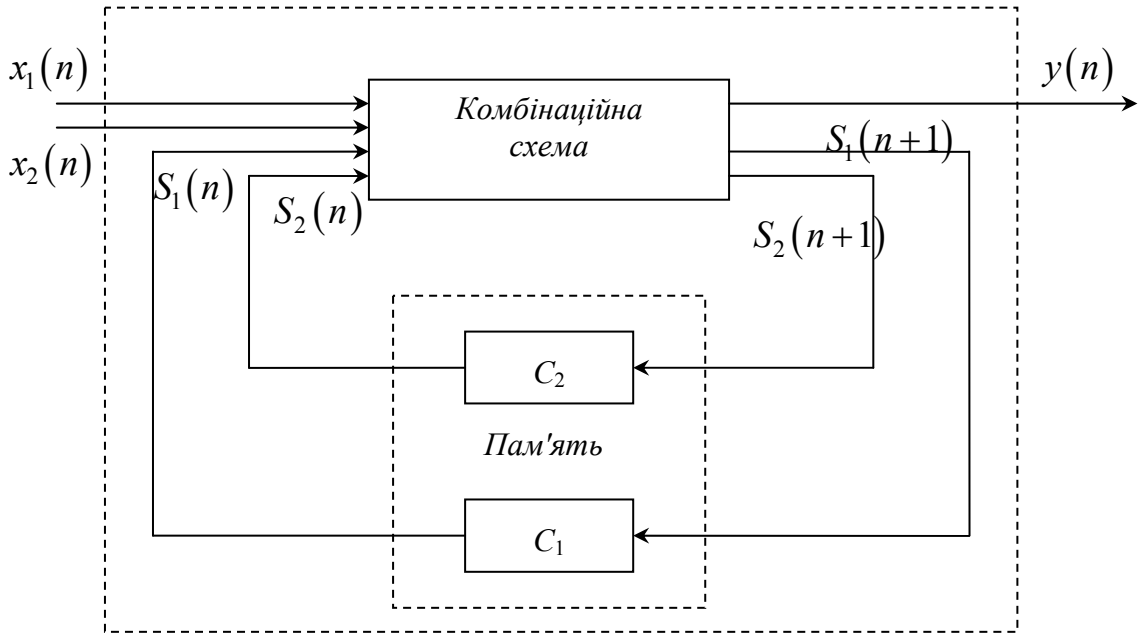


Рис. 3.11. Структурна схема скінченного автомата (приклад 15)

Виходячи з того, що праві частини логічних виразів для одержання функцій $s_1(n+1)$ й $s_2(n+1)$ містять загальний член $\overline{x_1}(n)\cdot\overline{x_1}(n)$, а $y(n)=\overline{s_1}(n+1)$, схема скінченного автомата може бути також подана у такому вигляді (див. рис. 3.12):

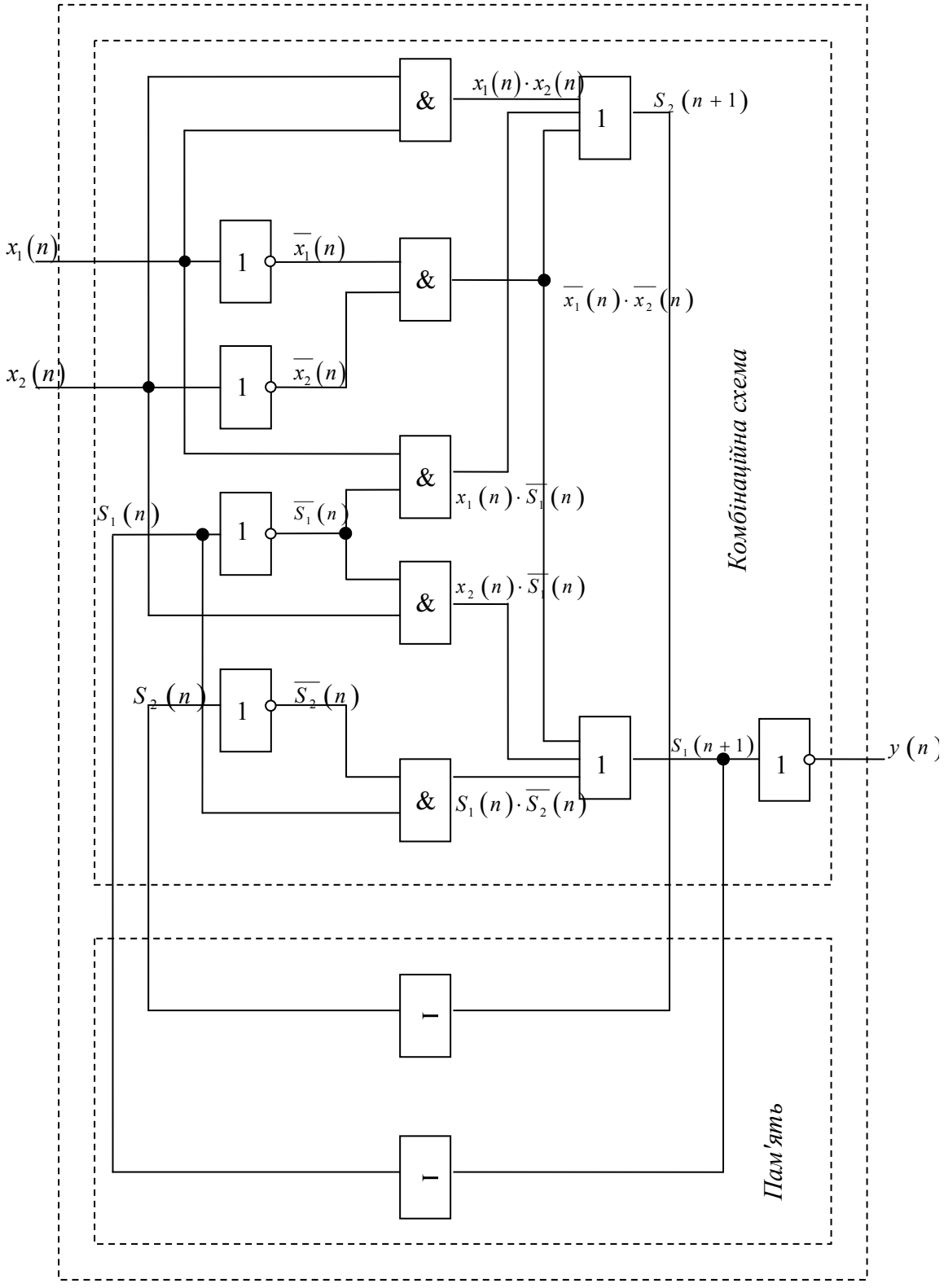


Рис. 3.12. Схема скінченного автомата (до прикладу 15)

Питання для самоконтролю

1. Яких значень можуть набувати булеві змінні й функції?
2. Яким чином визначають основні булеві функції однієї та двох змінних?
3. У якому випадку дві булеві функції будуть рівносильними?
4. Який запис булевих функцій називається досконалою диз'юнктивною (кон'юнктивною) нормальною формою?
5. Який закон алгебри логіки лежить в основі методів мінімізації булевих функцій?
6. Які методи мінімізації логічних функцій ви знаєте?
7. Що являє собою код Грея?
8. Що являє собою частково визначена логічна функція?
9. Який вигляд має загальна структура скінченного автомата?
10. У чому полягає відмінність скінченного автомата від комбінаційної схеми?

Задачі для самостійного розв'язування

1. Використовуючи основні тотожності булевої алгебри, записати дану функцію у вигляді досконалої диз'юнктивної нормальної форми.

$$y = \overline{x_1} \vee \left(\overline{x_2 \wedge x_3}\right) \vee x_1 \wedge \overline{x_3}\,.$$

2. Використовуючи основні тотожності булевої алгебри, записати дану функцію у вигляді досконалої кон'юнктивної нормальної форми.

$$y = \left(\overline{x_1 \wedge x_2}\right) \wedge \overline{x_3} x_2 \vee \overline{x_1 \wedge x_2}\,.$$

3. За допомогою таблиць істинності перевірити тотожність таких функцій: $y_1 = \left(x_1 \vee x_2\right) \wedge \overline{x_3}$ і $y_2 = \overline{\left(\overline{x_1} \wedge \overline{x_2}\right) \vee x_3}$.

4. Застосовуючи метод Квайна, знайти тупикову форму для диз'юнктивної нормальної форми такої функції:

$$f\left(x_1, x_2, x_3, x_4\right) = \overline{x_1} x_3 + \overline{x_2} x_3 x_4 + x_1 \overline{x_3}\,\overline{x_4} + \overline{x_1} x_2 x_3 \overline{x_4} + x_1 x_2 \overline{x_3} x_4 + x_1 \overline{x_2} x_3 x_4\,.$$

5. За допомогою методу Квайна–Мак-Класкі знайти тупикову ДНФ такої функції:

$$f\left(x_1, x_2, x_3, x_4\right) = x_2 x_4 + x_1 \overline{x_2} x_3 + x_1 \overline{x_3}\,\overline{x_4} + x_1 x_2 \overline{x_3} x_4 + \overline{x_1} x_2 \overline{x_3} x_4 + x_1 \overline{x_2}\,\overline{x_3} x_4\,.$$

6. Задану нижче логічну функцію записати у вигляді ДДНФ, а потім мінімізувати методом Вейча – Карно.

$$f\left(x_1,x_2,x_3,x_4\right)=\overline{x_1x_2}+x_1\overline{x_2}\left(\overline{x_3+x_2x_4}\right)+\overline{x_1}x_2x_4+x_1\overline{x_2x_3}x_4.$$

7. Методом Вейча – Карно знайти тупикову форму такої частково визначеної функції чотирьох змінних:

$$f\left(x_1,x_2,x_3,x_4\right)=\bigvee_1\left(1^*,2,3^*,5^*,6,7^*,8^*,9,11^*,12^*,13,14\right).$$

Тут символом «*» позначено десяткові номери наборів, на яких функцію не визначено.

8. Методом Вейча – Карно знайти тупикову форму записаної нижче функції чотирьох змінних й побудувати відповідну їй комбінаційну схему.

$$f\left(x_1,x_2,x_3x_4\right)=\bigvee_1\left(0,2,3,5,6,8,10,11,13,14\right).$$

9. На основі заданої таблиці істинності функції трьох змінних $f\left(x_1,x_2,x_3\right)$ скласти відповідну їй комбінаційну схему. Функцію $f\left(x_1,x_2,x_3\right)$ спочатку мінімізувати одним із відомих методів.

x_1	0	1	0	1	0	1	0	1
x_2	0	0	1	1	0	0	1	1
x_3	0	0	0	0	1	1	1	1
f	1	1	1	0	0	0	1	1

10. Синтезувати скінченний автомат, заданий загальною таблицею переходів.

$S(n)$ \ $X(n)$	0	1	2	3
0	2/1	3/0	2/1	2/1
1	2/1	3/0	1/1	1/0
2	2/1	1/1	0/0	0/0
3	2/1	1/1	0/1	0/1

Основні позначення

$\varnothing$ – порожня множина;

$x \in X$ – елемент x належить множині X;

$N = \{1, 2, 3, 4...\}$ – множина натуральних чисел;

$Z = \{...-3, -2, -1, 0, 1, 2, 3...\}$ – множина цілих чисел;

$Q = \left\{ \frac{g_1}{g_2} \middle| g_1 \in Z, g_2 \in N \right\}$ – множина раціональних чисел;

$R = \{x | a \le x \le b\}$ – множина дійсних чисел відрізка $(a;b)$;

$A \subset X$ підмножина A міститься в множині X;

S – універсальна множина;

$|X|$ – потужність (кардинальне число) множини X;

α_0 – потужність лічильної множини;

c – потужність континуальної множини;

$A \bigcup B$ – об'єднання множин;

$A \bigcap B$ – перетин множин;

$\overline{A}$ – доповнення множини;

$A - B$ або $\left(A \diagup B \right)$ – різниця множин;

$\overline{x}$ – заперечення;

$x_1 \vee x_2$ – диз'юнкція;

$x_1 \wedge x_2$ – кон'юнкція;

КНФ – кон'юнктивна нормальна форма;

ДНФ – диз'юнктивна нормальна форма;

ДКНФ – досконала кон'юнктивна нормальна форма;

ДДНФ – досконала диз'юнктивна нормальна форма;

Предметний покажчик

Список літератури

1. Гилл А. Введение в теорию конечных автоматов [Текст] / А. Гилл. – М. : Наука, 1966. – 272 с.
2. Глушков В.М. Синтез цифровых автоматов [Текст] / В.М. Глушков. – М. : Физматгиз, 1962. – 476 с.
3. Закревский А.Д. Алгоритмы синтеза дискретных устройств [Текст] / А.Д. Закревский. – М. : Наука, 1971. – 511с.
4. Кузин Л.Т. Основы кибернетики. Т. 2. Основы кибернетических моделей [Текст] / Л.Т. Кузин. – М. : Энергия, 1979. – 584 с.
5. Лазарев В.Г. Проектирование дискретных устройств автоматики [Текст] / В.Г. Лазарев, Н.П. Маркин, Ю.В. Лазарев. – М. : Радио и связь, 1985. – 168 с.
6. Майника Э. Алгоритмы оптимизации на сетях и графах [Текст] / Э. Майника. – М. : Мир, 1981. – 323 с.
7. Мелихов А.Н. Ориентированные графы и конечные автоматы [Текст] / А.Н. Мелихов. – М. : Наука, 1971. – 415 с.
8. Подлипенский В.С. Бесконтактные логические схемы автоматики: основы построения [Текст] / В.С. Подлипенский. – К. : 1965. – 216 с.
9. Поспелов Д.А. Логические методы анализа и синтеза схем [Текст] / Д.А. Поспелов. – М. : Энергия, 1974. – 368 с.
10. Сигорский В.П. Математический аппарат инженера [Текст] / В.П. Сигорский. – К. : Техніка, 1977. – 768 с.
11. Трачик, В. Дискретные устройства автоматики [Текст] / В. Трачик. – М. : Энергия, 1978. – 445 с.
12. Филлипс Д. Методы анализа сетей [Текст] / Д. Филлипс, А. Гарсиа-Диас. – М. : Мир, 1984. – 496 с.
13. Форд Л.Р. Потоки в сетях [Текст] / Л.Р. Форд, Д. Фалкерсон. – М. : Мир, 1966. – 276 с.
14. Уилсон Р. Введение в теорию графов [Текст] / Р. Уилсон. – М. : Мир, 1977. – 208 с.
15. Шоломов Л.А. Основы теории дискретных логических и вычислительных машин [Текст] / Л.А. Шоломов. – М. : Физматгиз, 1980. – 362 с.

Навчальне видання

Ігор Валерійович **Новицький**
Світлана Альбертівна **Ус**

ДИСКРЕТНА МАТЕМАТИКА

У ПРИКЛАДАХ І ЗАДАЧАХ

Навчальний посібник

Редактор О.Н. Ільченко

Підписано до друку 05.09.13. Формат 30х42/4.
Папір офсет. Ризографія. Ум. друк. арк. 5,0.
Обл.-вид. арк. 6,3. Тираж 50 пр. Зам. № .

Підготовлено до друку та видрукувано
в Державному вищому навчальному закладі
«Національний гірничий університет»
Свідоцтво про внесення до Державного реєстру ДК № 1842 від 11.06.2004 р.

49027, м. Дніпропетровськ, просп. К. Маркса,19.