МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА“

Кафедра систем штучного інтелекту

Лабораторна робота №3
З дисципліни
«Дискретна математика»

Виконала:
Студентка групи КН-115
Галік Вікторія
Викладач:
Мельникова Н. І.

Львів – 2019р.

Тема: Побудова матриці бінарного відношення

Мета роботи: набуття практичних вмінь та навичок при побудові матриць бінарних відношень та визначені їх типів.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Декартів добуток множин A і B (позначається A× B) – це множина всіх упорядкованих пар елементів (a,b), де a ∈ A, b∈ B. При цьому вважається, що (a1,b1) = (a2,b2) тоді і тільки тоді, коли a1 = a2 , b1 = b2.

Потужність декартова добутку дорівнює A× B = A × B .

Бінарним відношенням R називається підмножина декартового добутку A×B (тобто R ⊂ A×B).

Якщо пара (a,b) належить відношенню R , то пишуть (a, b)∈R , або aRb .

Областю визначення бінарного відношення R ⊂ X ×Y називається множина δ R = {x ∃y (x, y)∈R} , а областю значень – множина ρ R = {y ∃x (x, y)∈R} (∃- існує).

Для скінчених множин бінарне відношення R ⊂ A×B зручно задавати за допомогою матриці відношення Rm×n = (rij) , де m = A , a n = B .

Види бінарних відношень.

Нехай задано бінарне відношення R на множині A : R ⊆ A× A= {(a, b) a∈ A, b∈ A}

1. Бінарне відношення R на множині A називається рефлексивним, якщо для будь якого a ∈ A виконується aRa , тобто (a,a)∈R . Головна діагональ матриці рефлексивного відношення складається з одиниць. Граф рефлексивного відношення обов’язково має петлі у кожній вершині.

2. Бінарне відношення R на множині A називається антирефлексивним, якщо для будь якого a ∈ A не виконується aRa , тобто (a,a)∉ R . Головна діагональ матриці антирефлексивного відношення складається з нулів. Граф антирефлексивного відношення не має петель.

3. Бінарне відношення R на множині A називається симетричним, якщо для будь яких a,b∈ A з aRb слідує bRa , тобто якщо (a,b)∈R то і (b,a)∈ R . Матриця симетричного відношення симетрична відносно головної діагоналі. Граф симетричного відношення не є орієнтованим.

4. Бінарне відношення R на множині A називається антисиметричним, якщо для будь яких a,b∈ A з aRb та bRa слідує що a = b . Тобто якщо (a,b)∈R і (b,a)∈ R , то a = b . Матриця антисиметричного відношення не має жодної пари одиниць, які знаходяться на симетричних місцях по відношенню до головної діагоналі. У графа антисиметричного відношення вершини з’єднуються тільки однією напрямною дугою.

5. Бінарне відношення R на множині A називається транзитивним, якщо для будь яких a, b, c∈ A з aRb та bRc слідує, що aRc . Тобто якщо (a,b)∈R і (b,c)∈ R, то (a,c)∈ R . Матриця транзитивного відношення характеризується тим, що якщо елемент матриці σij = 1 та

σjm =1, то обов’язково σim =1. Граф транзитивного відношення такий, що якщо з’єднані дугами, наприклад, перша-друга та другатретя вершини, то обов’язково є дуга з першої в третю вершину.

6. Бінарне відношення R на множині A називається антитранзитивним, якщо для будь яких a, b, c∈ A з aRb та bRc слідує що не виконується aRc . Тобто якщо (a, b)∈R і (b, c)∈ R, то (a, c)∉ R . Матриця антитранзитивного відношення характеризується тим, що якщо елемент матриці σij = 1 та σjm =1, то обов’язково σim =0. Граф транзитивного відношення такий, що якщо з’єднані дугами, наприклад, перша-друга та друга-третя вершини, то обов’язково немає дуги з першої в третю вершину.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Варіант № 5

1. Чи є вірною рівність (A×B) ∩ (C×D) = (A×D) ∩ (C×B) ?

2. Знайти матрицю відношення :
$R = \{(x,\ y) | x \in M \ \&\ y \subset M \ \&\ |y| < x + 2\}$, де $M = \{x | x \in Z \ \&\ |x| \le 1\}$, Z - множина цілих чисел.

3. Зобразити відношення графічно: $\alpha = \{(x,\ y) | (x, y) \in R^2 \ \&\ (x + y)^2 = 4\}$, де R - множина дійсних чисел.

4. Навести приклад бінарного відношення R⊂A×A, де A={a,b,c,d,e}, яке є рефлексивне, несиметричне, транзитивне, та побудувати його матрицю.

5. Визначити множину (якщо це можливо), на якій дане відношення є:

а) функціональним; б) бієктивним:

$\alpha = \{(x,\ y) | (x, y) \in R^2 \ \& xy = 2\}$

РОЗВ’ЯЗАННЯ

1.

A * B = { (x, y) | (x ∈ A) ^ (y ∈ B) }   C * D = { (x, y) | (x ∈ C) ^ (y ∈ D) }

(A * B) ∩ (C * D) = { (x, y) | x ∈ (A ^ C) && y ∈ (B ^ D) }

A * D = { (x, y) | (x ∈ A) ^ (y ∈ D) }   C * B = { (x, y) | (x ∈ C) ^ (y ∈ B) }

(A * D) ∩ (C * B) = { (x, y) | x ∈ (A ^ C) && y ∈ (D ^ B) }

Рівність є вірною

2. $R = \{(x,\ y) | x \in M \ \&\ y \subset M \ \&\ |y| < x + 2\}$, де $M = \{x | x \in Z \ \&\ |x| \le 1\}$,

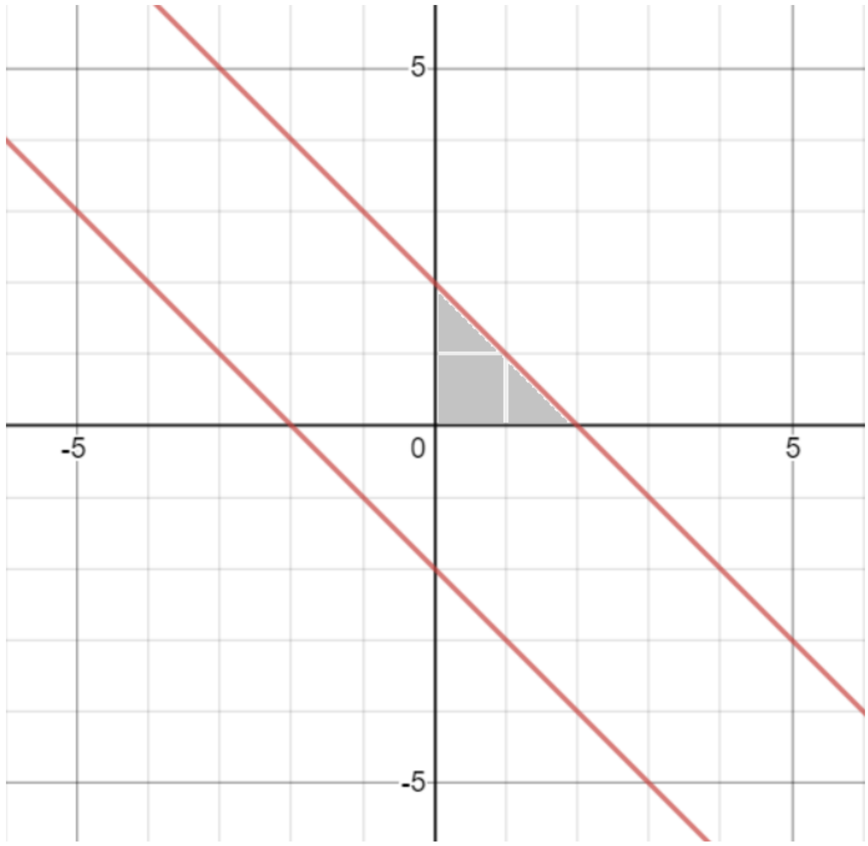
M = { ∅, { -1 }, { 0 }, { 1 } }

	∅	{ -1 }	{ 0 }	{ 1 }	{ -1, 0 }	{ -1, 1 }	{ 0, 1 }	{ -1, 0, 1 }
1	1	1	1	1	1	1	1	0
-1	1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0

3.

$\alpha = \{(x,\ y) | (x, y) \in R^2 \ \&\ (x + y)^2 = 4\}$,

$$\begin{cases} x + y = 2 \\ x + y = -2 \end{cases}$$



Зафарбована ділянка є відношенням, оскільки (x, y) ∈ R^2

4. R⊂A×A, де A={a,b,c,d,e}

Рефлексивне відношення : одиниці на головній діагоналі

Несиметричне відношення : $a_{ij} \neq a_{ji}$

Транзитивне : a_{ij} = a_{jk} = a_{ik} = 1

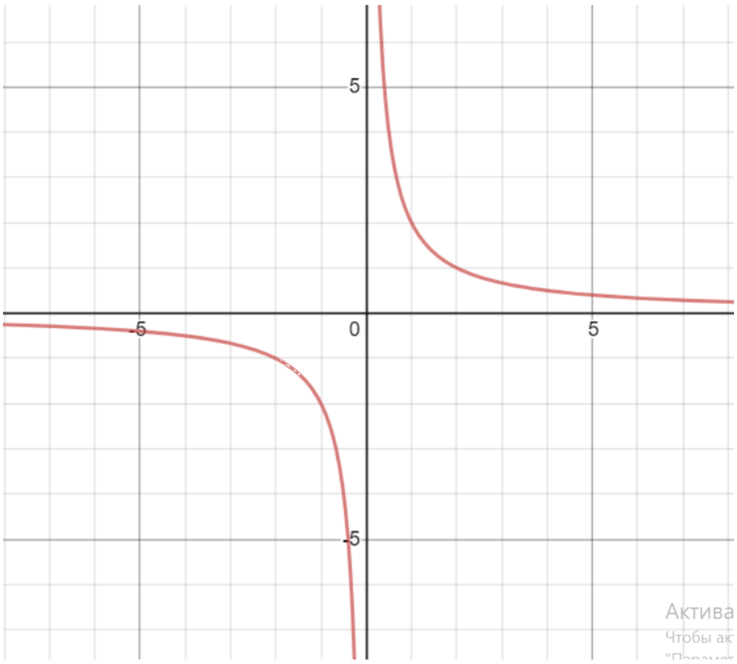
$$R = \begin{vmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{vmatrix}$$

5. $\alpha = \{(x,\ y) | (x, y) \in R^2\ \&\, xy = 2\}$

Зображення відношення зводиться до графічного розв'язку рівняння :

xy = 2

y = 2/x



D : x ∈ $(-\infty; 0) \cup (0; +\infty)$

E : y ∈ $(-\infty; 0) \cup (0; +\infty)$

Кожному значенню X відповідає одне і тільки одне значення У та кожному значенню У відповідає одне і тільки одне значення X.

Код програми:

```
Laaaaab3.cpp
Laaaaab3                                        (Global Scope)

#include <iostream>
#include <math.h>

using namespace std;

void Output(int arr[], int size);
int fact(int n);

int main() {
    int A[50];
    int size_a;
    int B[50];
    ///////////////// creation //////////////////
    cout << endl << "\t============CREATION_OF_SETS============" << endl << endl;

    cout << "\tChoose a size of sets (n*n) : ";
    cin >> size_a;
    //Check(size_a, 1, 50);
    do {
        if (size_a > 50 || size_a < 0) {
            cout << "  ERROR, try again :\t";
            cin >> size_a;
        }
    } while (size_a > 50 || size_a < 0);

    cout << "\tFill A-set of numbers : " << endl;
    for (int i = 0; i < size_a; i++) {
        cout << "\t";
        cin >> A[i];
    }
    cout << "    ..................................................." << endl;

    cout << "\tFill B-set of numbers : " << endl;
    for (int i = 0; i < size_a; i++) {
        cout << "\t";
        cin >> B[i];
    }
    /////////////// output //////////////////
    cout << "\t\t   YOUR_SETS_ARE : " << endl << endl;

35 %        No issues found
```

Laaaaab3.cpp

Laaaaab3 (Global Scope)

```cpp
    }
    //////////////// output //////////////////
    cout << "\t\t   YOUR_SETS_ARE : " << endl << endl;
    cout << "\t\tA = ";
    Output(A, size_a);
    cout << "\t\tB = ";
    Output(B, size_a);

    //////////// creation of relation ///////////////
    cout <<endl << "\t============BINARY_RELATION============" << endl;
    cout << endl << "\tCartesian  product of A and B ( A * B ) is :" << endl << endl;

    for (int i = 0; i < size_a; i++) {                              // output of product
        for (int j = 0;  j < size_a;  j++)
        {
            cout << "\t";
            cout << "(" << A[i] << ", " << B[j] << ")";
        }
        cout << endl;
    }
    cout << endl;
    cout << "    ................................................." << endl;
    cout << "\tCouples which satisfy the condition :" << endl;      // output of couples which satisfy the condition
    int counter = 0;
    for (int i = 0; i < size_a; i++) {
        for (int j = 0; j < size_a; j++)
        {
            if ((A[i] + 2) > 3 * B[j]) {
                cout << "(" << A[i] << ", " << B[j] << ")" << "    ";
            }
            else {
                counter++;
            }
        }
    }
    if (counter == size_a * size_a) {
        cout << " Couples are not found...";
    }
    cout << endl << endl;
```

Laaaaab3.cpp

Laaaaab3 (Global Scope)

```cpp
        cout << " Couples are not found...";
    }
    cout << endl << endl;
    ////////////// matrix of relation /////////////
    cout << "\t============MATRIX_OF_RELATION============" << endl << endl;
    int prod[100][100];
    for (int i = 0; i < size_a; i++)                // filling with appropriate values
    {
        for (int j = 0; j < size_a; j++)
        {
            if ((A[i] + 2) > 3 * B[j]) {
                prod[i][j] = 1;
            }
            else {
                prod[i][j] = 0;
            }

        }
    }

    for (int i = 0; i < size_a; i++)            // output of matrix
    {
        cout << "\t";
        for (int j = 0; j < size_a; j++)
        {
            cout << prod[i][j] << "    ";
        }
        cout << endl;
    }
    cout << endl << endl;
    //////// types of relation ///////////
    cout << "\t===========Types_of_relation============" << endl << endl;
    cout << "\tThis relation is :" << endl;
    if (counter == size_a * size_a) {
        cout << "\tEmpty";
    }
    else {
        /////////// reflexive ///////////////
        int ref = 0;
```

Laaaaab3.cpp

Laaaaab3 (Global Scope)

```cpp
else {
    /////////// reflexive //////////////
    int ref = 0;
    for (int i = 0; i < size_a; i++)
    {
        if (prod[i][i] == 1) {
            ref++;
        }
    }
    if (ref == size_a) {
        cout << "\t\tReflexive " << endl;
    }

    ///////////// antireflexive /////////////
    int aref = 0;
    for (int i = 0; i < size_a; i++)
    {
        if (prod[i][i] == 0) {
            aref++;
        }
    }
    if (aref == size_a) {
        cout << "\t\tAntireflexive " << endl;
    }

    ///////////// symmetrical ////////////////
    int sym = 0;
    for (int i = 0; i < size_a; i++)
    {
        for (int j = 0; j < size_a; j++)
        {
            if ((prod[i][j] == prod[j][i]) && i!=j ) {
                sym++;
            }
        }
    }
    if (sym == pow(size_a, 2) - size_a ) {
        cout << "\t\tSymmetrical" << endl;
    }
```

Laaaaab3.cpp

Laaaaab3 (Global Scope)

```cpp

///////// antisymmetrical ///////////////
int asym = 0;
for (int i = 0; i < size_a; i++)
{
    for (int j = 0; j < size_a; j++)
    {
        if (prod[i][j] == prod[j][i]) {
            asym++;
        }
    }
}

if (asym == 0) {
    cout << "\t\tAntisymmetrical" << endl;
}

//////////// transitive //////////////
int tranz = 0;

for (int i = 0; i < size_a; i++)
{
    for (int j = 0; j < size_a; j++)
    {
        if (prod[i][j] == 1)
        {
            for (int k = 0; k < size_a; k++)
            {
                if (prod[j][k] == 1)
                {
                    if (prod[i][k] == 1)
                    {
                        tranz = 1;
                    }
                    else {
                        tranz = 0;
                    }
                }
            }
```

Laaaaab3.cpp

Laaaaab3 (Global Scope)

```cpp

        if (tranz == 1) {
            cout << "\t\tTransitive" << endl;
        }
        if (tranz == 0) {
            cout << "\t\tNot transitive" << endl;
        }

        if (ref == size_a && sym == pow(size_a, 2) && tranz == 1)
        {
            cout << "\tThe type of relation is : " << endl <<"\t\tEquivalence" << endl;
        }
        if (ref == size_a && sym == pow(size_a, 2))
        {
            cout << "\tThe type of relation is : " << endl << "\t\tCompatibility" << endl;
        }
        if (ref == size_a && asym == 0 && tranz == 1)
        {
            cout << "\tThe type of relation is : " << endl << "\t\tPartial order" << endl;
        }
        if (aref == size_a && asym == 0 && tranz == 1)
        {
            cout << "\tThe type of relation is : " << endl << "\t\tFull order" << endl;
        }
    }
    cout << endl << endl;
    cout << "    ..................................................." << endl;

    system("pause");
    return 0;
}
void Output(int arr[], int size) {
    cout << " { ";
    for (int i = 0; i < size; i++) {
        if (i == (size - 1)) {
            cout << arr[i] << " ";
        }
        else {
            cout << arr[i] << ", ";
```

Результат програми :

D:\Desktop\Дискретна математика\Лабораторна №3\Laaaaab3\Debug\Laaaaab3.exe

```
        ============CREATION_OF_SETS============

        Choose a size of sets (n*n) : 4
        Fill A-set of numbers :
        2
        0
        9
        7
    ..................................................
        Fill B-set of numbers :
        6
        3
        1
        0
                   YOUR_SETS_ARE :

                A =  { 2, 0, 9, 7 }
                B =  { 6, 3, 1, 0 }

        ============BINARY_RELATION============

        Cartesian  product of A and B ( A * B ) is :

        (2, 6)  (2, 3)  (2, 1)  (2, 0)
        (0, 6)  (0, 3)  (0, 1)  (0, 0)
        (9, 6)  (9, 3)  (9, 1)  (9, 0)
        (7, 6)  (7, 3)  (7, 1)  (7, 0)

    ..................................................
```

```
    ..................................................
        Couples which satisfy the condition :
(2, 1)    (2, 0)    (0, 0)    (9, 3)    (9, 1)    (9, 0)    (7, 1)    (7, 0)

        ============MATRIX_OF_RELATION============

        0    0    1    1
        0    0    0    1
        0    1    1    1
        0    0    1    1


        ===========Types_of_relation============

        This relation is :
                Transitive
```

```
D:\Desktop\Дискретна математика\Лабораторна №3\Laaaaab3\Debug\Laaaaab3.exe

        ============CREATION_OF_SETS============

        Choose a size of sets (n*n) : 6
        Fill A-set of numbers :
        0
        7
        9
        8
        2
        4
    ..................................................
        Fill B-set of numbers :
        6
        9
        3
        1
        0
        2
                   YOUR_SETS_ARE :

                A =  { 0, 7, 9, 8, 2, 4 }
                B =  { 6, 9, 3, 1, 0, 2 }

        ============BINARY_RELATION============

        Cartesian  product of A and B ( A * B ) is :

        (0, 6)  (0, 9)  (0, 3)  (0, 1)  (0, 0)  (0, 2)
        (7, 6)  (7, 9)  (7, 3)  (7, 1)  (7, 0)  (7, 2)
        (9, 6)  (9, 9)  (9, 3)  (9, 1)  (9, 0)  (9, 2)
        (8, 6)  (8, 9)  (8, 3)  (8, 1)  (8, 0)  (8, 2)
        (2, 6)  (2, 9)  (2, 3)  (2, 1)  (2, 0)  (2, 2)
        (4, 6)  (4, 9)  (4, 3)  (4, 1)  (4, 0)  (4, 2)

    ..................................................
        Couples which satisfy the condition :
(0, 0)    (7, 1)    (7, 0)    (7, 2)    (9, 3)    (9, 1)    (9, 0)    (9, 2)    (8, 3)    (8, 1)    (8, 0)    (8, 2)
(2, 1)    (2, 0)    (4, 1)    (4, 0)
```

```
        ============MATRIX_OF_RELATION============

        0     0     0     0     1     0
        0     0     0     1     1     1
        0     0     1     1     1     1
        0     0     1     1     1     1
        0     0     0     1     1     0
        0     0     0     1     1     0


        ===========Types_of_relation============

        This relation is :
                Transitive


    ..................................................
```

Висновок : ми набули практичних вмінь та навичок при побудові матриць бінарних відношень та визначені їх типів, програмно побудували матрицю відношення та перевірили якого вона виду.