МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту

Лабораторна робота №3

З дисципліни «Дискретна математика»

Побудова матриці бінарного відношення

Виконав:

студент групи КН-110

Єлечкл Олег Андрійович

Викладач:

Мельникова Наталія Іванівна

Львів – 2018р.

Мета: набуття практичних вмінь та навичок при побудові матриць бінарних відношень та визначенні їхніх типів.

1. Практична частина

1.1 Перевірити рівність

$$(A \cup B) \times (C \cup D) = (A \times C) \cup (B \times C) \cup (A \times D) \cup (B \times D)$$

Щоби перевірити цю рівність, складемо предикат множини, що утворюється в результаті обчислень правої та лівої частини:

$(A \cup B) \times (C \cup D) = \{(x, y) \mid x \in (A \cup B) \ \& \ y \in (C \cup D)\}$ (1)
$(A \times C) \cup (B \times D) = \{(x, y) \mid x \in A \ \& \ y \in C\} \cup \{(x, y) \mid x \in B \ \& \ y \in D\} = \{(x, y) \mid x \in A \ \& \ y \in C \ \& \ x \in B \ \& \ y \in D\}$, при цьому, якщо $x \in A \ \& \ x \in B$, то отже $x \in (A \cup B)$, аналогічно $y \in (C \cup D)$. Звідси випливає, що $(A \times C) \cup (B \times C) \cup (A \times D) \cup (B \times D) = \{(x, y) \mid x \in (A \cup B) \ \& \ y \in (C \cup D)\}$, а це те ж саме, що і у формулі
(1).

Отже рівність справджується.

1.2 Знайти матрицю відношення

Потрібно знайти матрицю відношення $R \subset 2^A \times 2^B$, $R = \{(x, y)\ x \subset A \& \ y \subset B \& \ y \subset x\}$, де $A = \{2,4\}$, $B = \{1,2,4\}$.

За означенням множини А, $2^A = \{\emptyset, \{2\}, \{4\}, \{2,4\}\}$.
За означенням множини В, $2^B = \{\emptyset, \{1\}, \{2\}, \{4\}, \{1,2\}, \{1,4\}, \{2,4\}, \{1,2,4\}\}$.
Матриця, що задовольнить всі умови, виглядатиме так:

2^A \ 2^B	$\emptyset$	{1}	{2}	{4}	{1,2}	{1,4}	{2,4}	{1,2,4}
$\emptyset$	0	0	0	0	0	0	0	0

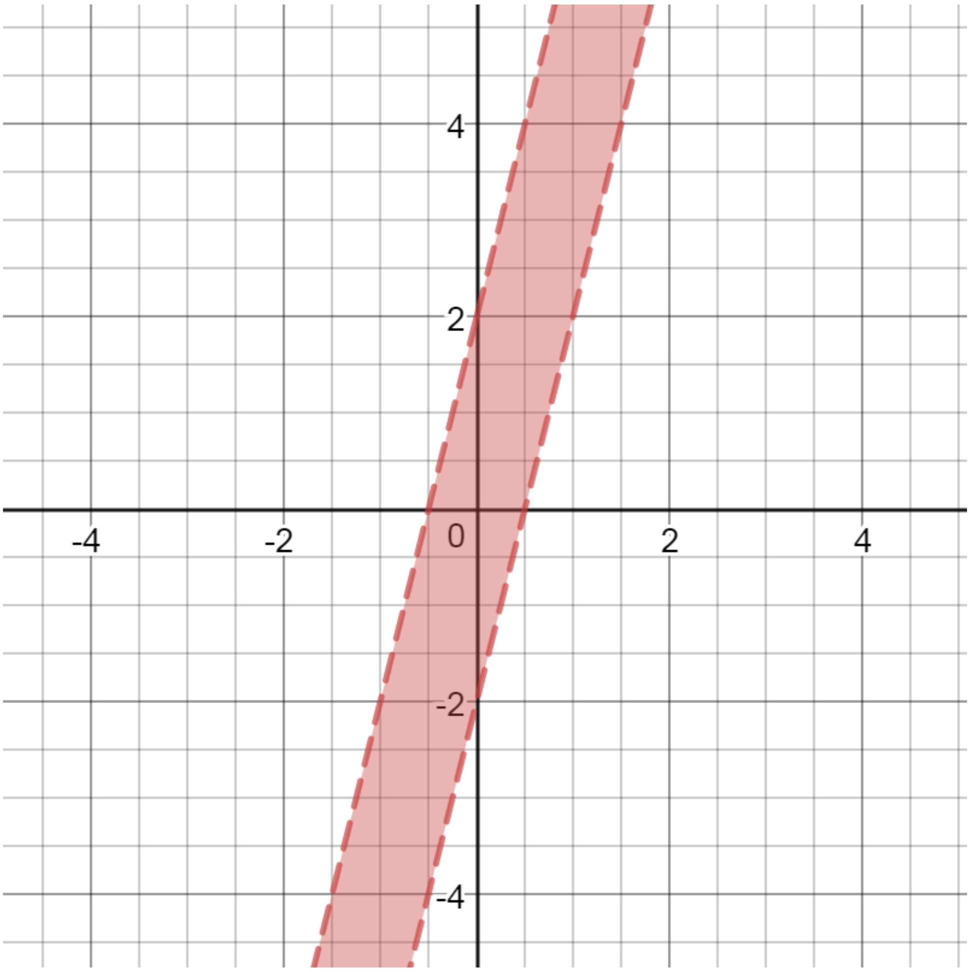
{2}	0	0	0	0	1	0	1	1
{4}	0	0	0	0	0	1	1	1
{2,4}	0	0	0	0	0	0	0	1

1.3 Зобразити відношення графічно

$\alpha = \{(x, y) | (x, y) \in R^2 \,\&\, | y - 4x | < 2$, де R - множина дійсних чисел.

$$\begin{cases} -y + 4x < 2 \\ y - 4x < 2 \end{cases} \Leftrightarrow \begin{cases} y > 4x - 2 \\ y < 4x + 2 \end{cases}$$

Ця система нерівностей породжує такий графік:



1.4 Побудувати матрицю для різних видів відношення

3. Маємо бінарне відношення $R \subset A \times A$, де $A = \{a,b,c,d,e\}$, яке задане своєю матрицею:

$$A(R) = \begin{matrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{matrix}$$

.Перевірити чи є дане відношення рефлексивним, симетричним, транзитивним, антисиметричним?
Дане відношення є рефлексивне так як головна діагональ дорівнює 1, не симетричним так як {2,4} ≠{4,2} і тд., і не транзитивне так як {2,4} ={4,5}≠{5,2}.

1.5 Визначити множину (якщо це можливо), на якій дане відношення є: а) функціональним; б) бієктивним:

$\alpha = \{(x,y\}|(x,y)\} \in R^2 \ \& \ y = e^{x-1}\}$

Задано відношення $\alpha = \{(x,y\}|(x,y)\} \in R^2 \ \& \ y = e^{-1}\}$. Перепишемо рівняння $y = e^{x-1}$ його відносно x :

$\ln(y) = x-1 \Rightarrow x = \ln(y)+1$
З поданого рівняння випливає, що $(x, y) \in \alpha \Rightarrow x \in (-\infty;\infty)$.

Графічним представленням такого відношення буде графік показникової функції яка перетинає вісь Оу у точці$(0;e^{-1})$.

Оскільки кожному значенню y відповідає значення x, дане відношення є функціональним коли $\alpha \subset \{(0; \infty)\}$. Аналогічно, тільки на цій же множині задане відношення може бути бієктивним.

2. Комп'ютерна програма

Потрібно написати програму, що знаходить матрицю бінарного відношення $\alpha \subset A \times B$, заданого на двох числових множинах, що потрібно вводити вручну. Програма має виводити на екран матрицю відношення та перевіряти його тип. Її код виглядає так:

```
#include <stdio.h>
#include <math.h>

int main(){

    /* Enter arrays A and B */
    int size;
    printf("Enter size of arrays: ");
    scanf("%d", &size);
    printf("Enter array A\n");
    int A[size];
    for(int i=0; i<size; i++){
        printf("A[%d] = ", i);
        scanf("%d", &A[i]);
    }
    int B[size];
    printf("\nEnter array B\n");
    for(int i=0; i<size; i++){
        printf("B[%d] = ", i);
        scanf("%d", &B[i]);
    }
    printf("\nA = ");
    for(int i=0; i<size; i++){
        printf("%d ", A[i]);
```

```
}
printf("\n");
printf("B = ");
for(int i=0; i<size; i++){
    printf("%d ", B[i]);
}
printf("\n");


/* Generate the binary matrix */
int matr[size][size];
for(int r=0; r<size; r++){
    for(int c=0; c<size; c++){
        if(A[r] <( 2*B[c] + 1))
            matr[r][c] = 1;
        else
            matr[r][c] = 0;
    }
}


/* Print generated binary matrix */
printf("\n");
for(int r=0; r<size; r++){
```

```
    for(int c=0; c<size; c++){
        printf("%d ", matr[r][c]);
        if(c==size-1)
            printf("\n");
    }
}
/* Generate the inverse matrix */
int antimatr[size][size];
for(int r=0; r<size; r++){
    for(int c=0; c<size; c++){
        antimatr[r][c] = matr[c][r];
    }
}
/* Check for 0 */
int flag7 = 0;
for(int r=0; r<size; r++){
    for(int c=0; c<size; c++){
        if(matr[r][c]== 1)
            flag7 = 1;
    }
```

```
}

/* Check for reflexivity */
printf("\n");
int flag1=1;
for(int i=0; i<size; i++){
    if (matr[i][i]==0)
        flag1=0;
}
if(flag1==1)
    printf("This relation is reflexive.\n");
else if(flag1==0)
    printf("This relation is not reflexive.\n");

/* Check for antireflexivity */
int flag2=0;
for(int i=0; i<size; i++){
    if (matr[i][i]==1)
        flag2=1;
}
if(flag2==0)
```

```
        printf("This relation is antireflexive.\n");
    else if(flag2==1)
        printf("This relation is not antireflexive.\n");


    /* Check for symmetry */
    int flag3=1;
    for(int r=0; r<size; r++){
        for(int c=0; c<size; c++){
            if((r!=c) && (antimatr[r][c]!=matr[r][c]))
                flag3 = 0;
        }
    }

    if(flag3==1)
        printf("This relation is symmetric\n");
    else if(flag3==0)
        printf("This relation is not symmetric\n");


    /* Check for antisymmetry */
    int flag4=0;
    for(int r=0; r<size; r++){
        for(int c=0; c<size; c++){
```

```
            if((r!=c) && (antimatr[r][c]==matr[r][c]))
                flag4 = 1;
        }
    }
    if(flag4==0)
        printf("This relation is antisymmetric\n");
    else if(flag4==1)
        printf("This relation is not antisymmetric\n");

    /* Check for transitivity */
    int Asize = pow(size, 3);
    int Aflag[Asize];
    int ai=0;
    for(int i=0; i<size; i++){
        for(int j=0; j<size; j++){
            for(int k=0; k<size; k++){
                if((i!=j && i!=k && j!=k) && (matr[i][j] == 0 || matr[j][k] == 0))
                    Aflag[ai]=1;
                else if((i!=j && i!=k && j!=k) && (matr[i][j] == 1 && matr[j][k] == 1 && matr[i][k] == 1))
                    Aflag[ai]=1;
                else if ((i!=j && i!=k && j!=k) && (matr[i][j] == 1 && matr[j][k] == 1 && matr[i][k] == 0))
```

```
                Aflag[ai]=0;
            else
                Aflag[ai]=1;
            ai++;
        }
    }
}
int flag5 = 1;
for (int i=0; i<Asize; i++){
    if(Aflag[i]==0)
        flag5=0;
}
if(flag7 == 0)
    flag5 = 0;
if(flag5==1)
    printf("This relation is transitive\n");
else if(flag5==0)
    printf("This relation is not transitive\n");


/* Check for antitransitivity */
int Bsize = pow(size, 3);
int Bflag[Bsize];
int bi=0;
```

```
for(int i=0; i<size; i++){
    for(int j=0; j<size; j++){
        for(int k=0; k<size; k++){
            if((i!=j && i!=k && j!=k) && (matr[i][j] == 0 || matr[j][k] == 0))
                Bflag[bi]=1;
            else if((i!=j && i!=k && j!=k) && (matr[i][j] == 1 && matr[j][k] == 1 && matr[i][k] == 1))
                Bflag[bi]=0;
            else if ((i!=j && i!=k && j!=k) && (matr[i][j] == 1 && matr[j][k] == 1 && matr[i][k] == 0))
                Bflag[bi]=1;
            else
                Bflag[bi]=1;
            bi++;
        }
    }
}
int flag6 = 1;
for (int i=0; i<Bsize; i++){
    if(Bflag[i]==0)
        flag6=0;
}
if(flag7 == 0 || flag5==1)
    flag6 = 0;
```

```
    if(flag6==1)
        printf("This relation is antitransitive\n");
    else if(flag6==0)
        printf("This relation is not antitransitive\n");

    printf("\n");
}
```

Висновок :

На цій лабораторній роботі я набув практичних вмінь з побудови матриць бінарних відношень та визначенні їхніх типів.