

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теоретических основ
компьютерной безопасности и
криптографии

Классификация бинарных отношений и системы замыканий

ОТЧЁТ

ПО ДИСЦИПЛИНЕ

«ПРИКЛАДНАЯ УНИВЕРСАЛЬНАЯ АЛГЕБРА»

студентки 3 курса 331 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Шуликиной Анастасии Александровны

Преподаватель

профессор, д.ф.-м.н.

подпись, дата

В. А. Молчанов

Саратов 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 Цель работы и порядок её выполнения	4
2 Теория	5
2.1 Виды и классификация бинарных отношений	5
2.1.1 Отношение эквивалентности	5
2.1.2 Отношение квазипорядка	6
2.1.3 Отношение частичного порядка	6
2.2 Системы замыкания на множестве бинарных отношений	6
2.2.1 Алгоритмы построения основных замыканий бинарных отношений	7
3 Программная реализация рассмотренных алгоритмов	9
3.1 Результаты тестирования программы	9
3.2 Код программы, на основе рассмотренных алгоритмов, на языке C++	9

ВВЕДЕНИЕ

В данной лабораторной работе поставлена задача рассмотрения основных свойств бинарных отношений, их классификация и замыкание, а также написание алгоритмов для определения классификации и замыкания бинарного отношения.

1 Цель работы и порядок её выполнения

Цель работы – изучение основных свойств бинарных отношений и операций замыкания бинарных отношений.

Порядок выполнения работы:

1. Разобрать основные определения видов бинарных отношений и разработать алгоритмы классификации бинарных отношений.
2. Изучить свойства бинарных отношений и рассмотреть основные системы замыкания на множестве бинарных отношений.
3. Разработать алгоритмы построения основных замыканий бинарных отношений.

2 Теория

2.1 Виды и классификация бинарных отношений

Бинарным отношением между элементами A и B называется любое подмножество ρ множества $A \times B$, то есть $\rho \subset A \times B$.

По определению, бинарным отношением называется множество пар. Если ρ – бинарное отношение (т.е. множество пар), то говорят, что параметры x и y связаны бинарным отношением ρ , если пара $\langle x, y \rangle$ является элементом ρ , т.е. $\langle x, y \rangle \in \rho$.

Бинарное отношение $\rho \subset A \times B$ называется:

- рефлексивным, если $(a, a) \in \rho$ для любого $a \in A$;
- симметричным, если $(a, b) \in \rho \Rightarrow (b, a) \in \rho$;
- антисимметричным, если $(a, b) \in \rho$ и $(b, a) \in \rho \Rightarrow a = b$;
- транзитивным, если $(a, b) \in \rho$ и $(b, c) \in \rho \Rightarrow (a, c) \in \rho$.

Существует три основных типа бинарных отношений:

- отношение эквивалентности
- отношение квазипорядка
- отношение частичного порядка

2.1.1 Отношение эквивалентности

Бинарное отношение ε на множестве A называют отношением эквивалентности, если оно рефлексивно, симметрично и транзитивно.

Алгоритм проверки отношения на эквивалентность:

1. Из определения, для того, чтобы бинарное отношение являлось отношением эквивалентности, оно должно включать в себя свойства рефлексивности, симметричности и транзитивности. Поэтому первым делом производится проверка на рефлексивность, симметричность и транзитивность.
2. Выполняется операция $\&$ всех результатов.
3. Производится проверка на истинность или ложность. Если полученное значение – истинно, то отношение является отношением эквивалентности, если ложно, то отношение отношением эквивалентности не является.

2.1.2 Отношение квазипорядка

Бинарное отношение ε на множестве A называют отношением квазипорядка, если оно рефлексивно и транзитивно.

Алгоритм проверки отношения на отношение квазипорядка:

1. Из определения, для того, чтобы бинарное отношение являлось отношением квазипорядка, оно должно включать в себя свойства рефлексивности и транзитивности. Поэтому первым делом производится проверка на рефлексивность и транзитивность.
2. Выполняется операция $\&$ всех результатов.
3. Производится проверка на истинность или ложность. Если получившееся значение – истинно, то отношение является отношением квазипорядка, если ложно, то отношение отношением квазипорядка не является.

2.1.3 Отношение частичного порядка

Бинарное отношение ε на множестве A называют отношением частичного порядка, если оно рефлексивно, антисимметрично и транзитивно.

Алгоритм проверки отношения на отношение частичного порядка:

1. Из определения, для того, чтобы бинарное отношение являлось отношением частичного порядка, оно должно включать в себя свойства рефлексивности, антисимметричности и транзитивности. Поэтому первым делом производится проверка на рефлексивность, антисимметричность и транзитивность.
2. Выполняется операция $\&$ всех результатов.
3. Производится проверка на истинность или ложность. Если получившееся значение – истинно, то отношение является отношением частичного порядка, если ложно, то отношение отношением частичного порядка не является.

2.2 Системы замыкания на множестве бинарных отношений

Замыканием отношения R относительно свойства P называется такое множество R^* , что:

1. $R \subset R^*$
2. R^* обладает свойством P .
3. R^* является подмножеством любого другого отношения, содержащего R и обладающего свойством P .

То есть R^* является минимальным надмножеством множества R , выдерживается P .

Множество Z подмножеств множества A называется системой замыканий, если оно замкнуто относительно пересечений, т.е. выполняется $\cap B \in Z$ для любого подмножества $B \subset Z$.

В частности, для $\emptyset \subset Z$ выполняется $\cap \emptyset = A \in Z$

На множестве всех бинарных отношений между элементами множества A^2 следующие множества являются системами замыканий:

1. Z_r – множество всех рефлексивных бинарных отношений между элементами множества A ,
2. Z_s – множество всех симметричных бинарных отношений между элементами множества A ,
3. Z_t – множество всех транзитивных бинарных отношений между элементами множества A ,
4. $Z_{eq} = Eq(A)$ – множество всех отношений эквивалентности на множестве A .

Множество Z_{as} всех антисимметричных бинарных отношений между элементами множества A не является системой замыкания.

На множестве $P(A^2)$ всех бинарных отношений между элементами множества A следующие отображения являются операторами замыканий:

1. $f_r(\rho) = \rho \cup \Delta_A$ – наименьшее рефлексивное бинарное отношение, содержащее отношение $\rho \subset A^2$,
2. $f_s(\rho) = \rho \cup \rho^{-1}$ – наименьшее симметричное бинарное отношение, содержащее отношение $\rho \subset A^2$,
3. $f_t(\rho) = \cup_{n=1}^{\infty} \rho^n$ – наименьшее транзитивное бинарное отношение, содержащее отношение $\rho \subset A^2$,
4. $f_{eq}(\rho) = f_t f_s f_r(\rho)$ – наименьшее отношение эквивалентности, содержащее отношение $\rho \subset A^2$.

2.2.1 Алгоритмы построения основных замыканий бинарных отношений

Построение замыкания отношения относительно свойствам рефлексивности.

1. Для начала необходимо создать пустой список для хранения пар замыканий.

2. Задать цикл i от 1 до N . Если $M_{ii} = 0$, пару (i, i) добавить в замыкание рефлексивности и замыкание эквивалентности.
3. Это замыкание бинарного отношения ρ относительно рефлексивности. Трудоёмкость $O(N)$.

Построение замыкания отношения относительно свойствам симметричности.

1. Для начала необходимо создать пустой список для хранения пар замыканий.
2. Задать цикл i от 1 до N и цикл по j от 1 до N . Если $M_{ii} = 0$ и $M_{ji} = 0$, добавить пару (j, i) добавить в замыкание симметричности и замыкание эквивалентности.
3. Это замыкание бинарного отношения ρ относительно симметричности. Трудоёмкость $O(N^2)$.

Построение замыкания отношения относительно свойствам транзитивности.

1. Для начала необходимо создать копию матрицы исходного бинарного отношения.
2. Задать цикл n от 1 до N , цикл k от 1 до N , цикл i от 1 до N и цикл по j от 1 до N . Если $M_{ki} = M_{ij} = 1$ и $M_{ki} = 0$, добавить пару (k, k) добавить в замыкание транзитивности и замыкание эквивалентности.
3. Это замыкание бинарного отношения ρ относительно транзитивности. Трудоёмкость $O(N^4)$.

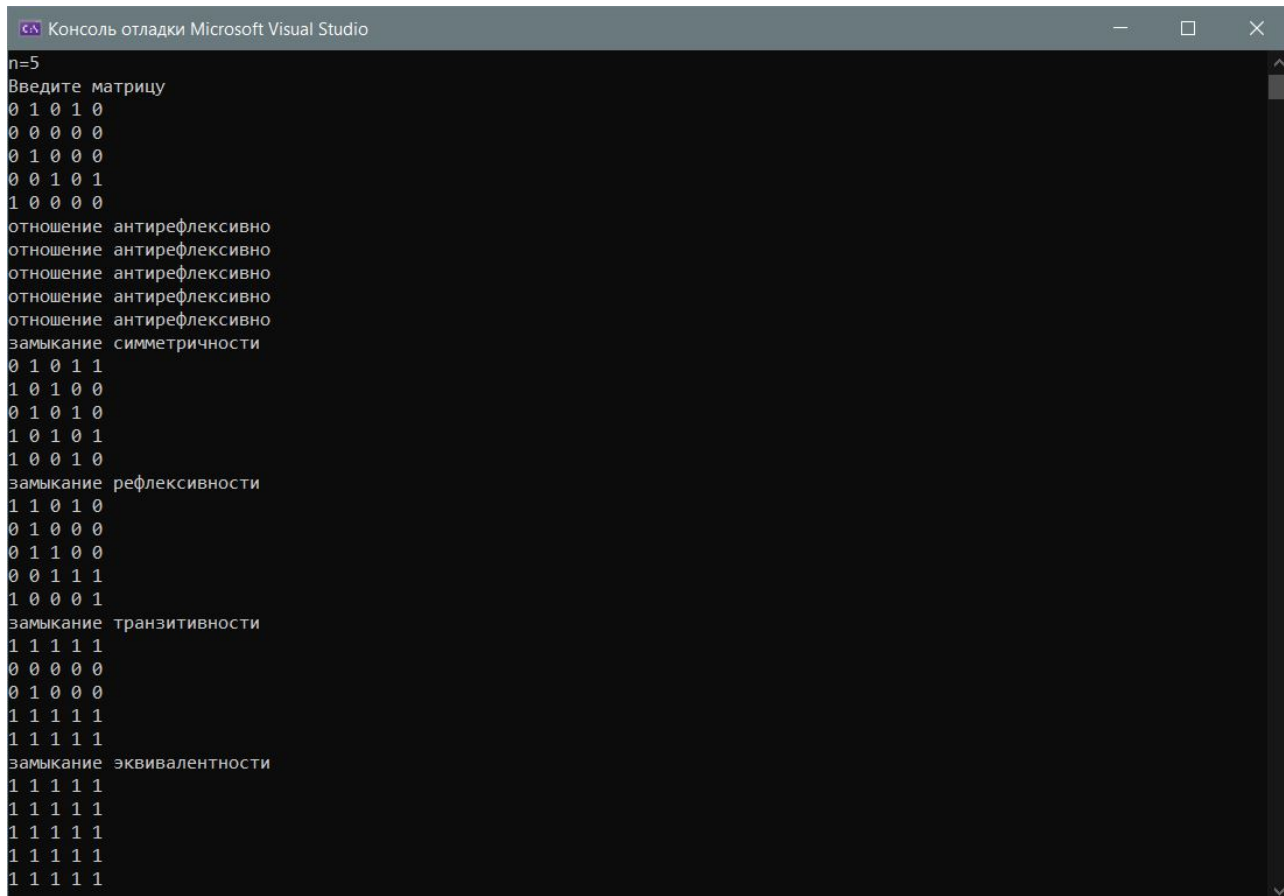
Построение замыкания отношения относительно свойствам эквивалентности.

1. Необходимо по очереди вызвать алгоритмы построения замыкания рефлексивности, симметричности и транзитивности.
2. Это эквивалентное замыкание бинарного отношения ρ . Трудоёмкость $O(N^4) = O(N + N^2 + N^4)$.

3 Программная реализация рассмотренных алгоритмов

3.1 Результаты тестирования программы

На рисунке 1 можно увидеть работу, реализуемой программы, по рассмотренным алгоритмам.



```
Консоль отладки Microsoft Visual Studio
n=5
Введите матрицу
0 1 0 1 0
0 0 0 0 0
0 1 0 0 0
0 0 1 0 1
1 0 0 0 0
отношение антирефлексивно
отношение антирефлексивно
отношение антирефлексивно
отношение антирефлексивно
отношение антирефлексивно
замыкание симметричности
0 1 0 1 1
1 0 1 0 0
0 1 0 1 0
1 0 1 0 1
1 0 0 1 0
замыкание рефлексивности
1 1 0 1 0
0 1 0 0 0
0 1 1 0 0
0 0 1 1 1
1 0 0 0 1
замыкание транзитивности
1 1 1 1 1
0 0 0 0 0
0 1 0 0 0
1 1 1 1 1
1 1 1 1 1
замыкание эквивалентности
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
```

Рисунок 1 – Тест программы №1

3.2 Код программы, на основе рассмотренных алгоритмов, на языке C++

```
#include <iostream>
```

```
using namespace std;
```

```
int symmetry1 = 0, reflexivity1 = 0, transitivity1 = 0;
```

```
void symmetry(int** a, int n)
```

```
{
```

```
    int antisymmetry1 = 0;
```

```

for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        if (a[i][j] != a[j][i])
        {
            antisymmetry1++;
        }
        else symmetry1++;
    }
}

if (symmetry1 == 1)
cout << "relation is symmetric" << endl;
else if (antisymmetry1 == 1)
cout << "relation is antisymmetric" << endl;
}

void reflexivity(int** a, int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (a[i][i] == 1)
                reflexivity1++;
        }
        if (reflexivity1 >= 1) {
            cout << "relation is reflexive" << endl;
        }
        else
            cout << "relation is antireflexive" << endl;
    }
}

```

```

}

void transitivity(int** a, int n)
{
    int not_transitivity1 = 0;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (a[i][j])
            {
                for (int k = 0; k < n; k++)
                {
                    if (a[j][k] && !a[i][k])
                    {
                        not_transitivity1++;
                    }
                    else
                    {
                        transitivity1++;
                    }
                }
            }
        }
    }

    if (not_transitivity1 == 1)
        cout << "relation is not transitive" << endl;
    else if (transitivity1 == 1)
        cout << "relation is transitive" << endl;
}

```

```

void pr_symmetry(int** a, int n)
{
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            if (a[i][j] == 1)
                a[j][i] = 1;
}

void pr_reflexivity(int** a, int n)
{
    for (int i = 0; i < n; i++)
        a[i][i] = 1;
}

void pr_transitivity(int** a, int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            for (int k = 0; k < n; k++)
            {
                if (a[j][k] == 1)
                {
                    for (int p = 0; p < n; p++)
                    {
                        if (a[k][p] == 1)
                            a[j][p] = 1;
                    }
                }
            }
        }
    }
}

```

```

}

void pr(int** a, int n, int number)
{
    int** a1;
    a1 = new int* [n];
    for (int i = 0; i < n; i++)
    {
        a1[i] = new int[n];
        for (int j = 0; j < n; j++)
        {
            a1[i][j] = a[i][j];
        }
    }
    if (number == 1)
    {
        pr_symmetry(a1, n);
        cout << "symmetric closure" << endl;
    }
    if (number == 2)
    {
        pr_reflexivity(a1, n);
        cout << "reflexivity closure" << endl;
    }
    if (number == 3)
    {
        pr_transitivity(a1, n);
        cout << "transitive closure" << endl;
    }
    if (number == 4)
    {
        pr_symmetry(a1, n);
        pr_reflexivity(a1, n);
        pr_transitivity(a1, n);
    }
}

```

```

        cout << "equivalence closure" << endl;
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++)
            cout << a1[i][j] << ' ';
        cout << endl;
    }
}

```

```

int main()
{
    setlocale(LC_ALL, "RUS");
    int n;
    cout << "n=";
    cin >> n;
    int** a;
    a = new int* [n];
    cout << "Enter matrix \n";
    for (int i = 0; i < n; i++)
    {
        a[i] = new int[n];
        for (int j = 0; j < n; j++)
        {
            cin >> a[i][j];
        }
    }
    symmetry(a, n);
    reflexivity(a, n);
    transitivity(a, n);

    if (symmetry1 == 1 && reflexivity1 == 1
        && transitivity1 == 1)

```

```

    cout << "The relation is an
              equivalence relation" << endl;
    if (reflexivity1 == 1 && transitivity1 == 1)
    cout << "The relation is a
              quasi-order relation" << endl;
    if (symmetry1 == 0 && reflexivity1 == 1
        && transitivity1 == 1)
    cout << "The relation is a
              partial order relation" << endl;
    pr(a, n, 1);
    pr(a, n, 2);
    pr(a, n, 3);
    pr(a, n, 4);

    return 0;
}

```

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы были рассмотрены основные свойства бинарных отношений, виды бинарных отношений, при определённых комбинациях свойств, а также изучена система замыканий на множестве бинарных отношений. Также были разработаны алгоритмы определения свойств отношений и их классификации.