МИНОБРНАУКИ РОССИИ
ФГБОУ ВО «СГУ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

КЛАССИФИКАЦИЯ БИНАРНЫХ ОТНОШЕНИЙ И СИСТЕМЫ ЗАМЫКАНИЙ

ЛАБОРАТОРНАЯ РАБОТА

студента 3 курса 331 группы
специальности 100501 — Компьютерная безопасность
факультета КНиИТ
Окунькова Сергея Викторовича

Проверил
аспирант _____ В. Н. Кутин

Саратов 2022

СОДЕРЖАНИЕ

1 Постановка задачи 3
2 Теоретические сведения по рассмотренным темам с их обоснованием . . . 4
3 Результаты работы 7
3.1 Описание алгоритма классификации бинарных отношений 7
3.2 Описание алгоритмов построения основных замыканий бинарных отношений 9
3.3 Коды программ, реализующей рассмотренные алгоритмы 10
3.4 Результаты тестирования программ 15
3.5 Оценки сложности рассмотренных алгоритмов 17
3.5.1 Алгоритм определения рефлексивности 17
3.5.2 Алгоритм определения антирефлексивности 17
3.5.3 Алгоритм определения симметричности 17
3.5.4 Алгоритм определения антисимметричности 17
3.5.5 Алгоритм определения транзитивности 18
3.5.6 Алгоритм классификации 18
3.5.7 Построение замыкания рефлексивности 18
3.5.8 Построение замыкания симметричности 18
3.5.9 Построение замыкания транзитивности 18
ЗАКЛЮЧЕНИЕ 19

1 Постановка задачи

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 Теоретические сведения по рассмотренным темам с их обоснованием

Подмножества декартова произведения $A \times B$ множеств A и B называются **бинарными отношениями** между элементами множеств A, B и обозначаются строчными греческими буквами: $\rho, \sigma, \rho_1, \rho_2, ...$

Для бинарного отношения $\rho \subset A \times B$ область определения D_ρ и множество значений E_ρ определяется как подмножества соответствующих множеств A и B по следующим формулам:

$$D_\rho = \{a : (a, b) \in \rho \text{ для некоторого } b \in B\}$$

$$E_\rho = \{b : (a, b) \in \rho \text{ для некоторого } a \in A\}$$

Бинарное отношение $\rho \subset A \times A$ является:

1. *рефлексивным*, если $(a, a) \in \rho$ для любого $a \in A$;
2. *антирефлексивным*, если $(a, a) \notin \rho$ для любого $a \in A$;
3. *симметричным*, если $(a, b) \in \rho \Rightarrow (b, a) \in \rho$;
4. *антисимметричным*, если $(a, b) \in \rho$ и $(b, a) \in \rho \Rightarrow a = b$;
5. *транзитивным*, если $(a, b) \in \rho$ и $(b, c) \in \rho \Rightarrow (a, c) \in \rho$
6. *антитранзитивным*, если $(a, b) \in \rho$ и $(b, c) \in \rho \Rightarrow (a, c) \notin \rho$

Приведем теоретико-множественное истолкование свойств бинарных отношений и перефразируем данное истолкование на язык матриц:

Пусть задано бинарное отношение $\rho \subset A \times A$ и определена матрица данного бинарного отношения $M(\rho)$, тогда бинарное отношение ρ называется

1. *рефлексивным* тогда и только тогда, когда $\Delta_A \subset \rho$. Это означает, что бинарное отношение ρ рефлексивно, если $M(\rho) \geq E$, где E - единичная матрица. Если же матрица $M(\rho)$ несравнима с единичной матрицей, то бинарное отношение ρ не является рефлексивным;
2. *симметричным* тогда и только тогда, когда $\rho^{-1} \subset \rho$. Это означает, что бинарное отношение ρ симметрично, если $M(\rho) \geq M(\rho)^T$, где $M(\rho)^T$ – транспонированная матрица бинарного отношения ρ. Если же матрица $M(\rho)$ несравнима с $M(\rho)^T$, то бинарное отношение ρ не является симметричным;
3. *транзитивным* тогда и только тогда, когда $\rho\rho \subset \rho$. Это означает, что бинарное отношение ρ транзитивно, если $M(\rho)M(\rho) \leq M(\rho)$. 3.

4. *антисимметричным* тогда и только тогда, когда $\rho \cap \rho^{-1} \subset \Delta_A$. Это означает, что бинарное отношение ρ антисимметрично, если $E \geq M(\rho) \otimes M(\rho)^T$ (значения элементов вне главной диагонали матрицы $M(\rho) \otimes M(\rho)^T$ равны нулю), где $\otimes$ - операция поэлементного умножение матриц;

Классификация бинарных отношений напрямую определяются их свойствами.

1. Рефлексивное транзитивное отношение называется отношением квазипорядка.
2. Рефлексивное симметричное транзитивное отношение называется отношением эквивалентности.
3. Рефлексивное антисимметричное транзитивное отношение называется отношением (частичного) порядка.
4. Антирефлексивное, антисимметричное и транзитивное отношение называется отношением строгого порядка.

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

Оператором замыкания на множестве A называется отображение f множества всех подмножеств P(A) в себя, удовлетворяющее условиям:

1. $X \subset Y \Rightarrow f(X) \subset f(Y)$
2. $X \subset f(X)$
3. $ff(X) = f(X)$

для всех $X, Y \in P(A)$. Для подмножества $X \subset A$ значение f(X) называется **замыканием** подмножества X. Множество Z подмножеств множества A называется **системой замыканий**, если оно замкнуто относительно пересечений, т.е. выполняется $\cap B \in Z$ для любого подмножества $B \subset Z$.

Лемма 1. О системах замыканий бинарных отношений.

На множестве $P(A^2)$ всех бинарных отношений между элементами множества A 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 элемен-

тами 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.

Лемма 2. О замыканиях бинарных отношений.

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, на содержащее отношение $\rho \subset A^2$.

3 Результаты работы

3.1 Описание алгоритма классификации бинарных отношений

1. Алгоритм 1 - Проверка бинарного отношения на рефлексивность:
   Бинарное отношение называется рефлексивным тогда и только тогда, когда $\Delta_A \subset \rho$. Это означает, что бинарное отношение ρ рефлексивно, если $M(\rho) \geq E$, где E - единичная матрица. Если же матрица $M(\rho)$ несравнима с единичной матрицей, то бинарное отношение ρ не является рефлексивным;
   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$
   Выход: "Множество рефлексивно"или "Множество не рефлексивно"
   Шаг 1. Суммирование элементов на главной диагонали ($sum = \sum_{i=1}^{n} a[i][i]$).
   Шаг 2. Если $sum = n$, то отношение является рефлексивным, иначе не рефлексивным.
   Асимптотика $O(n)$.
2. Алгоритм 2 - Проверка бинарного отношения на симметричность:
   Бинарное отношение называется симметричным тогда и только тогда, когда $\rho^{-1} \subset \rho$. Это означает, что бинарное отношение ρ симметрично, если $M(\rho) \geq M(\rho)^T$, где $M(\rho)^T$ – транспонированная матрица бинарного отношения ρ. Если же матрица $M(\rho)$ несравнима с $M(\rho)^T$, то бинарное отношение ρ не является симметричным;
   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$
   Выход: "Множество симметрично"или "Множество не симметрично"
   Шаг 1. Транспонируем A, чтобы получить $B = A^T$ ($0 \leq i, j < n, b[i][j] \in B : b[i][j] = a[j][i]$).
   Шаг 2. Если $A = B$ ($b[i][j] \in B : b[i][j] = a[i][j]$, где $0 \leq i, j < n$), то бинарное отношение будет является симметричным, иначе отношение не симметрично.
   Асимптотика $O(n^2)$.
3. Алгоритм 3 - Проверка бинарного отношения на транзитивность:
   Бинарное отношение транзитивным тогда и только тогда, когда $\rho\rho \subset \rho$. Это означает, что бинарное отношение ρ транзитивно, если $M(\rho)M(\rho) \leq M(\rho)$.
   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$
   Выход: "Множество транзитивно"или "Множество не транзитивно"

На вход подается матрица бинарного отношения A.

Шаг 1. Получить матрицу $B = A^2$.

Шаг 2. Сравнить полученную и исходную матрицу.

Шаг 3. Если $B \leq A$ ($b[i][j] \in B : b[i][j] \leq a[i][j]$, где $0 \leq i, j < n$), то бинарное отношение транзитивно, иначе не транзитивным.

Асимптотика $O(n^3)$.

4. Алгоритм 4 - Проверка бинарного отношения на антирефлексивность:

   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$

   Выход: "Множество антирефлексивно"или "Множество не антирефлексивно"

   На вход подается матрица бинарного отоношения A.

   Шаг 1. Суммирование элементов на главной диагонали ($sum = \sum_{i=1}^{n} a[i][i]$).

   Шаг 2. Если $sum = 0$, то отношение является антирефлексивным, иначе не антирефлексивным.

   Асимптотика $O(n)$.

5. Алгоритм 5 - Проверка бинарного отношения на антисимметричность:

   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$

   Выход: "Множество антисимметрично"или "Множество не антисимметрично"

   Шаг 1. Транспонируем A, чтобы получить $C = A^T$.

   Шаг 2. Получим матрицу B поэлементным умножением матрицы A на C.

   Шаг 4. Если $b[i][j] = 0$, где $b[i][j]$ элемент матрицы B, $0 \leq i, j < n$ и $i \neq j$, то отношение является антисиметричным, иначе отношение не антисиметрично.

   Асимптотика $O(n^2)$.

6. Алгоитм 6 - Классификация бинарного отношения:

   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$

   Выход: «Бинарное отношение является отношением квазипорядка», «Бинарное отношение является отношением эквивалентности», «Бинарное отношение является отношением частичного порядка» или «Бинарное отношение является отношением строгого порядка».

   Шаг 1. Запустить алгоритмы 1 и 3 (проверки на рефлексивность и транзитивность), подав им на вход матрицу A. Если алгоритмы вернут значения «Бинарное отношение является рефлексивным» и «Бинарное отношение

является транзитивным», то вернуть значение «Бинарное отношение является отношением квазипорядка».

Шаг 2. Запустить алгоритмы 1, 2 и 3 (проверки на рефлексивность, симметричность и транзитивность), подав им на вход матрицу A. Если алгоритмы вернут значения «Бинарное отношение является рефлексивным», «Бинарное отношение является симметричным» и «Бинарное отношение является транзитивным», то вернуть значение «Бинарное отношение является отношением эквивалентности».

Шаг 3. Запустить алгоритмы 3 и 5 (проверки на антисимметричность и транзитивность), подав им на вход матрицу A. Если алгоритмы вернут значения «Бинарное отношение является антисимметричным» и «Бинарное отношение является транзитивным», то вернуть значение «Бинарное отношение является отношением частичного порядка».

Шаг 4. Запустить алгоритмы 3, 4 и 5 (проверки на антирефлексивность, антисимметричность и транзитивность), подав им на вход матрицу A. Если алгоритмы вернут значения «Бинарное отношение является антирефлексивным», «Бинарное отношение является антисимметричным» и Бинарное отношение является транзитивным», то вернуть значение «Бинарное отношение является отношением строгого порядка».

Если не учитывать сложность вызываемых алгоритмов, то асимптотика $O(1)$, иначе асимптотика $O(n^3)$.

3.2 Описание алгоритмов построения основных замыканий бинарных отношений

1. Алгоритм 7 - Замыкание бинарного отношения относительно рефлексивности:

   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$

   Выход: матрица бинарного отношения A', замкнутая относительно рефлексивности

   Шаг 1. Присвоить каждому $a[i][i]$ значение 1, где $0 \leq i < n$: , после чего вернуть полученную матрицу бинарного отношения $A' = (a'_{ij})$ с построенным на нем рефлексивным замыканием.

   Асимптотика $O(n)$.

2. Алгоритм 8 - Замыкание бинарного отношения относительно симметричности:

Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$

Выход: матрица бинарного отношения A', замкнутая относительно симметричности

Шаг1. Каждому элементу $a[i][j]$ $0 \leq i, j < n$ матрицы A присваивается значение элемента $a[j][i]$, после чего вернуть полученную матрицу бинарного отношения $A' = (a'_{ij})$ с построенным на нем симметричным замыканием.

Асимптотика $O(n^2)$.

3. Алгоритм 9 - Замыкание бинарного отношения относительно транзитивности:

   Вход: матрица бинарного отношения $A = (a_{ij})$, размерности $n \times n$

   Выход: матрица бинарного отношения A', замкнутая относительно транзитивности

   Шаг1. Если $a[i][k] = 1$ и $a[k][j] = 1$, то присвоить $a[i][j]$ значение 1, где $0 \leq i, j, k < k$. Такой шаг нужно повторить n раз в силу определения п.3) оператора транзитивного замыкания в лемме 2, после чего вернуть полученную матрицу бинарного отношения $A' = (a'_{ij})$ с построенным на нем транзитивным замыканием.

   Асимптотика $O(n^4)$.

3.3 Коды программ, реализующей рассмотренные алгоритмы

```
import numpy as np
import json


def isTran(a):
    n = len(a)
    b = np.matmul(a, a)
    for i in range(n):
        for j in range(n):
            if b[i][j]:
                b[i][j] = b[i][j] / b[i][j]
    f = (a >= b).all()
    if f:
        print("Set is transitive")
        return 't'
    else:
        f1 = True
```

```
        for i in range(n):
            for j in range(n):
                for k in range(n):
                    if a[i][k] and a[k][j] and a[i][j]:
                        f1 = False
        if f1:
            print("Set is anti-transitive")
            return 'at'
        else:
            print("Set is not transitive")
            return 'nt'


def isSymm(a):
    b = a.transpose()
    if np.array_equal(a, b):
        print("Set is symmetry")
        return 's'
    else:
        f = True
        b = np.multiply(a, b)
        for i in range(len(b)):
            for j in range(len(b[i])):
                if b[i][j] != 0 and i != j:
                    f = False
                    break
        if f:
            print("Set is anti-symmetry")
            return 'as'
        else:
            print("Set is not symmetry")
            return 'ns'


def isRefl(a):
    n = len(a)
    sum = 0
    for i in range(n):
        sum += a[i][i]
    if sum == n:
        print("Set is reflexive")
```

```
        return 'r'
    elif sum == 0:
        print("Set is anti-reflexive")
        return 'ar'
    else:
        print("Set is not reflexive")
        return 'nr'


def makeRefl(a):
    n = len(a)
    for i in range(n):
        a[i][i] = 1


def makeSymm(a):
    n = len(a)
    for i in range(n):
        for j in range(n):
            if a[i][j]:
                a[j][i] = 1


def makeTran(a):
    n = len(a)
    for c in range(n):
        for k in range(n):
            for i in range(n):
                for j in range(n):
                    if a[i][k] and a[k][j]:
                        a[i][j] = 1

def get_set(a):
    s = []
    for i in range(len(a)):
        for j in range(len(a[i])):
            if a[i][j] == 1:
                s.append((i + 1, j + 1))
    return s
```

```
if __name__ == "__main__":
    print("How you want enter your relation (set or matrix)?")
    enter = input()
    if enter == 'set':
        print('Enter the number of elements in relation')
        n = int(input())
        print("Enter your set")
        # st = [(1, 3), (3, 4), (1, 4), (2, 5), (5, 3)]
        s = list(map(str, input().split(' ')))
        st = []
        for c in s:
            a = ''
            i = 1
            while c[i] != ',':
                a += c[i]
                i += 1
            i += 1
            b = ''
            while c[i] != ')':
                b += c[i]
                i += 1
            st.append((int(a), int(b)))
        a = np.zeros((n, n), int)
        for s in st:
            a[s[0] - 1][s[1] - 1] = 1
        print('----------------------------------------')
        print("Relation's matrix")
        print(a)
    else:
        print('Enter the number of elements in relation')
        n = int(input())
        print("Enter your matrix")
        a = [list(map(int, input().split())) for i in range(n)]
        a = np.array(a).reshape(n, n)
        print('----------------------------------------')
        print("Relation's set")
        print(*get_set(a))
    print('----------------------------------------')
    properties = {"reflexive": isRefl(a), "symmetry": isSymm(a), "transitive": isTran
    print('----------------------------------------')
    if properties['reflexive'] == 'r' and properties['symmetry'] == 's' and properties
```

```
    print('Relation is equivalent')
    print('----------------------------------------')
if properties['reflexive'] == 'r' and properties['transitive'] == 't':
    print('Relation is the relation of the quasi-order')
    print('----------------------------------------')
if properties['reflexive'] == 'r' and properties['symmetry'] == 'as' and properti
    print('Relation is the relation of the partial order')
    print('----------------------------------------')
if properties['reflexive'] == 'ar' and properties['symmetry'] == 'as' and properti
    print('Relation is the relation of the strict order')
    print('----------------------------------------')
if properties['reflexive'] != 'r':
    makeRefl(a)
if properties['symmetry'] != 's':
    makeSymm(a)
if properties['transitive'] != 't':
    makeTran(a)
print('Closure matrix: ')
print(a)
print('----------------------------------------')
print('Closure set: ')
print(*get_set(a))
print('----------------------------------------')
```

3.4 Результаты тестирования программ

```
How you want enter your relation (set or matrix)?
set
Enter the number of elements in relation
3
Enter your set
(1,2) (1,3)
----------------------------------------
Relation's matrix
[[0 1 1]
 [0 0 0]
 [0 0 0]]
----------------------------------------
Set is anti-reflexive
Set is anti-symmetry
Set is transitive
----------------------------------------
Relation is the relation of the strict order
----------------------------------------
Closure matrix:
[[1 1 1]
 [1 1 0]
 [1 0 1]]
----------------------------------------
Closure set:
(1, 1) (1, 2) (1, 3) (2, 1) (2, 2) (3, 1) (3, 3)
----------------------------------------
```

Рисунок 1 – Ввод бинарного отношения (1, 2), (1, 3) в виде множества с выводом свойств этого множества и замыкания относительно эквивалентности

```
How you want enter your relation (set or matrix)?
set
Enter the number of elements in relation
5
Enter your set
(3,2) (2,5) (4,2)
----------------------------------------
Relation's matrix
[[0 0 0 0 0]
 [0 0 0 0 1]
 [0 1 0 0 0]
 [0 1 0 0 0]
 [0 0 0 0 0]]
----------------------------------------
Set is anti-reflexive
Set is not symmetry
Set is anti-transitive
----------------------------------------
Closure matrix:
[[1 0 0 0 0]
 [0 1 1 1 1]
 [0 1 1 1 1]
 [0 1 1 1 1]
 [0 1 1 1 1]]
----------------------------------------
Closure set:
(1, 1) (2, 2) (2, 3) (2, 4) (2, 5) (3, 2) (3, 3) (3, 4) (3, 5) (4, 2) (4, 3) (4, 4) (4, 5) (5, 2) (5, 3) (5, 4) (5, 5)
----------------------------------------
```

Рисунок 2 – Ввод бинарного отношения (3, 2), (2, 5), (4, 2) в виде множества с выводом свойств этого множества и замыкания относительно эквивалентности

```
How you want enter your relation (set or matrix)?
matrix
Enter the number of elements in relation
5
Enter your matrix
0 0 0 0 0
0 0 0 0 1
0 1 0 0 0
0 1 0 0 0
0 0 0 0 0
----------------------------------------
Relation's set
(2, 5) (3, 2) (4, 2)
----------------------------------------
Set is anti-reflexive
Set is not symmetry
Set is anti-transitive
----------------------------------------
Closure matrix:
[[1 0 0 0 0]
 [0 1 1 1 1]
 [0 1 1 1 1]
 [0 1 1 1 1]
 [0 1 1 1 1]]
----------------------------------------
Closure set:
(1, 1) (2, 2) (2, 3) (2, 4) (2, 5) (3, 2) (3, 3) (3, 4) (3, 5) (4, 2) (4, 3) (4, 4) (4, 5) (5, 2) (5, 3) (5, 4) (5, 5)
----------------------------------------
```

Рисунок 3 – Ввод бинарного отношения (3, 2), (2, 5), (4, 2) в виде матрицы с выводом свойств этого множества и замыкания относительно эквивалентности

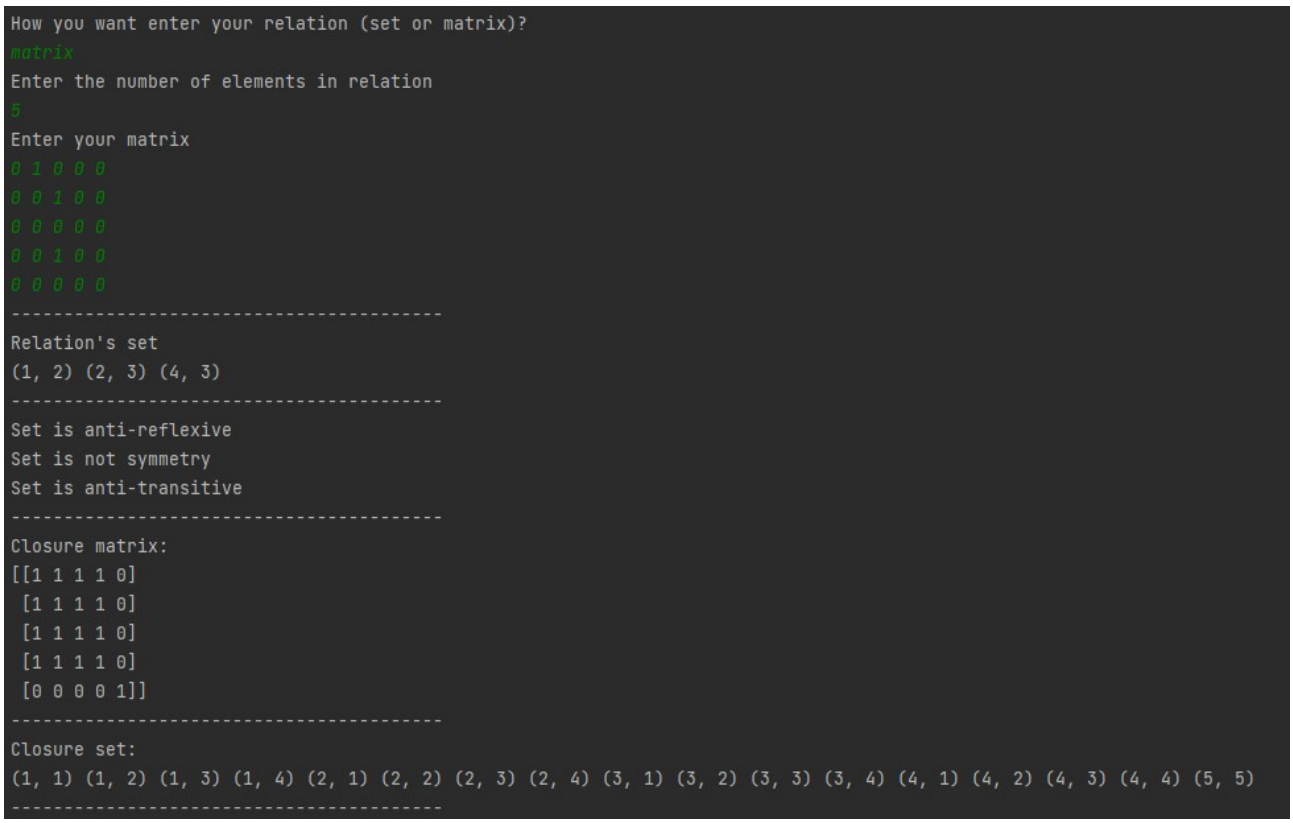

Рисунок 4 – Ввод бинарного отношения (1, 2), (2, 3), (4, 3) в виде матрицы с выводом свойств этого множества и замыкания относительно эквивалентности

3.5 Оценки сложности рассмотренных алгоритмов

3.5.1 Алгоритм определения рефлексивности

Сложность выполнения проверки на рефлексивность определяется как $O(n)$.

3.5.2 Алгоритм определения антирефлексивности

За счет схожести с алгоритмом определения рефлексивности сложность определяется как $O(n)$.

3.5.3 Алгоритм определения симметричности

Сложность транспонирования в numpy определяется как $O(n^{3/2}log\ n)$, сложность сравнение двух матриц поэлементно определяется как $O(n^2)$. Отсюда можно сделать вывод, что сложность алгоритма будет определятся как $O(n^{3/2}log\ n + n^2) = O(n^2)$.

3.5.4 Алгоритм определения антисимметричности

Сложность поэлементного умножения матриц определяется как $O(n^2)$, сложность сравнение двух матриц поэлементно определяется как $O(n^2)$. Отсюда

можно сделать вывод, что сложность будет определятся как $O(n^2+n^2) = O(n^2)$

3.5.5 Алгоритм определения транзитивности

Из всего выше сказанного очевидно, что сложность проверки на транзитивность или антитранзитивность составляет $O(n^3)$, так как в нем используется умножение, сравнение матриц и тройной цикл для проверки рефлексивности в худшем случае матриц.

3.5.6 Алгоритм классификации

Сложность выполнения самого алгоритма классификации бинарных отношений реализованно через словарь языка python и оператор if, поэтому является константной ($O(1)$), если не учитывать сложность выполнения проверки свойств отношения. Если учитывать сложность алгоритмов проверки свойств отношения, то :

1. Сложность проверка на квазипорядок определяется как $O(n^3 + n) = O(n^3)$.
2. Сложность проверка на эквивалентность определяется как $O(n^3 + n + n^{3/2} log\ n) = O(n^3)$.
3. Сложность проверка на частичный порядок определяется как $O(n^3 + n + n^3) = O(n^3)$.
4. Сложность проверка на строгий порядок определяется как $O(n^3 + n + n^3) = O(n^3)$.

3.5.7 Построение замыкания рефлексивности

Так как весь алгоритм строится на заполнении главной диагонали матрицы 1, то его сложность состовляет $O(n)$.

3.5.8 Построение замыкания симметричности

Для посторения замыкания симметричности используются вложенный цикла, поэтому сложность алгоритма определяется как $O(n^2)$.

3.5.9 Построение замыкания транзитивности

Для посторения замыкания транзитивности используются три вложенный цикла, поэтому сложность алгоритма определяется как $O(n^4)$.

ЗАКЛЮЧЕНИЕ

В рамках данной лабораторной работы были рассмотренны теоритические основы свойств бинарных отношений, их видов и методов их замыкания по каждому из свойств. На основе этой теоретической части была смоделирована программа на языке Python с использованием средств библиотеки Numpy, которая способна определить свойства заданного множества, его вид и построить систему замыкания по каждому из основных свойств бинарного отношения, а так же была оценена асимптотика каждого реализованного алгоритма.