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»

студента 3 курса 331 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Алексеева Александра Александровича

|  |  |  |
| --- | --- | --- |
| Преподаватель  профессор, д.ф.-м.н. | \_\_\_\_\_\_\_\_\_\_ | В. А. Молчанов |
|  | подпись, дата |  |

Саратов 2022

СОДЕРЖАНИЕ

1 Цель работы и порядок её выполнения…………………………………….3

2 Теория………………………………………………………………………..4

2.1 Виды и классификации бинарных отношений……………………4

2.2 Отношение эквивалентности………………………………………4

2.3 Отношение квазипорядка………………………………………….5

2.4 Отношение частичного порядка…………………………………...5

2.5 Системы замыкания на множестве бинарных отношений……….6

2.6 Алгоритм построения основных замыканий бин. отношений…...7

3 Результаты работы……………………………………………………………8

3.1 Оценки сложности рассмотренных алгоритмов………………….8

3.2 Результаты тестирования программ………………………………8

3.3 Код программы……………………………………………………11

ЗАКЛЮЧЕНИЕ……………………………………………………………….14

1 Цель работы и порядок её выполнения

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

Порядок выполненных 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 классификации бинарных отношений

Бинарным отношением между элементами A и B называется любое подмножество ρ множества A × B, то есть ρ ⊂ A × 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〈x, y〉является элементом ρ, т.е. 〈x, y〉 ∈ ρ.

Бинарное отношение ρ ⊂ A × B называется:

– рефлексивным, если (a, a) ∈ ρ для любого a ∈ A;

– симметричным, если (a, b) ∈ ρ ⇒ (b, a) ∈ ρ;

– антисимметричным, если (a, b) ∈ ρ и (b, a) ∈ ρ ⇒ a = b;

– транзитивным, если (a, b) ∈ ρ è (b, c) ∈ ρ ⇒ (a, c) ∈ ρ.

Существует три типа бинарных отношений:

– отношение эквивалентности

– отношение квазипорядка

– отношение частичного порядка

2.2 Отношение эквивалентности

Бинарное отношение ε на множестве А 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 операции & всех результатов.

3. Производится проверка на истинности или ложность. Если получившееся значение истинности, то отношение является отношением эквивалентности, если ложно, то отношение отношением эквивалентности не является.

2.3 Отношение квазипорядка

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

Алгоритм проверки отношения на эквивалентность:

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.

2.4 Отношение частичного порядка

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

Алгоритм проверка 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.

2.5 Системы замыкания на множестве бинарных отношений

Множество ℤ подмножеств множества А называется *системой замыканий*, если оно замкнуто относительно пересечений, т.е. выполняется

*B*ℤ для любого подмножества *B* ℤ*.*

Множества, принадлежащие системе замыканий ℤ, называются *замкнутыми подмножествами* множества А.

По теореме о полной решётке система замыканий является полной решёткой относительно теоретика-множественного включения. При этом точные грани в решётке (ℤ,⊂) могут отличаться от точных граней (пересечения и объединения) булеан *P*(A).

Лемма о системах замыканий бинарных отношений. На множестве *P*(A2) всех бинарных отношений между элементами множества А следующие множества являются системами замыканий:

1) ℤr – множество всех рефлексивных бинарных отношений между элементами множества А,

2) ℤs – множество всех симметричных бинарных отношений между элементами множества А,

3) ℤt – множество всех транзитивных бинарных отношений между элементами множества А,

4) ℤeq = *Eq*(A) – множество всех отношений эквивалентности на множестве А.

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

1) *X* *Y* ⇒ *f(X)* *f(Y);*

2) *X* *f(X);*

3) *ff(X) = f(X),*

для всех *X, Y* *P(A).*

Для подмножества *X* ⊂ *A* значение *f(X)* называется *замыканием* подмножества *X.*

2.6 Алгоритм построения основных замыканий бинарных отношений

Для того, чтобы построить замыкание:

1) рефлексивности для отношения ρ ⊂ A2,необходимо взять отношение ρ и добавить к нему все пары *(a, a)* ⊂ A2, *a,b* ∈ A.

2) симметричности для отношения ρ ⊂ A2,необходимо взять отношение ρ и если есть пара *(a, b)* ⊂ A2, то добавить пару *(b, a)* ⊂ A2, *a,b* ∈ A.

3) транзитивности для отношения ρ ⊂ A2, необходимо взять отношение ρ и если есть пары *(a, b)* и *(b, c)* ⊂ A2, то добавить пару *(a, c)* ⊂ A2, *a,b,c* ∈ A.

4) эквивалентности для отношения ρ ⊂ A2, необходимо взять отношение ρ и последовательно построить замыкание рефлексивности, симметричности и транзитивности.

3 Результаты работы

3.1 Оценки сложности рассмотренных алгоритмов

Проверка на рефлексивность – О(N);

Проверка на симметричность – О(N2/2) ~ О(N2);

Проверка на антисимметричность – О(N2/2) ~ О(N2);

Проверка на транзитивность – О(N3);

Построение замыкания рефлексивности (без вывода матрицы) – О(N);

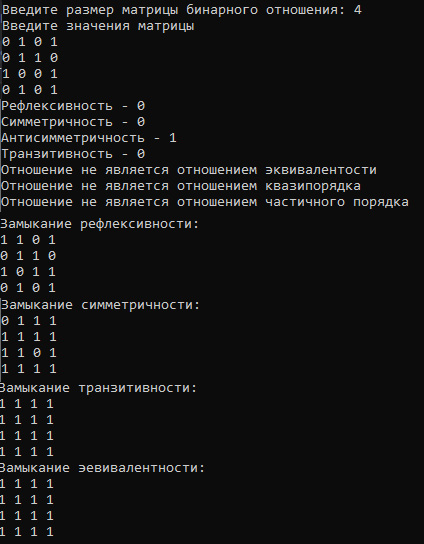
Построение замыкания симметричности – О(N2 + N2) ~ О(N2);

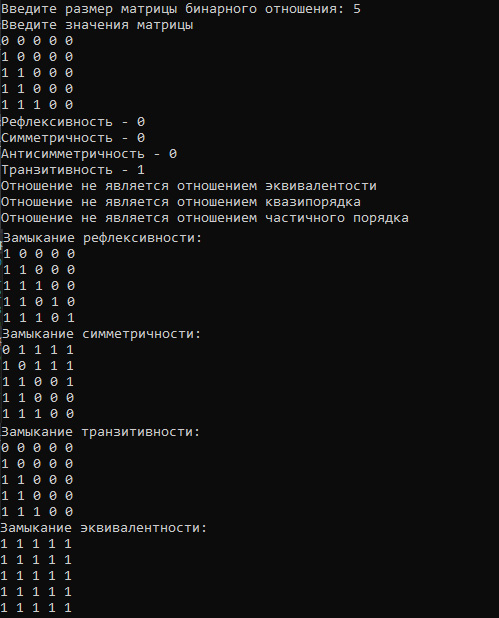
Построение замыкания транзитивности – О(N4 + N2) ~ О(N4);

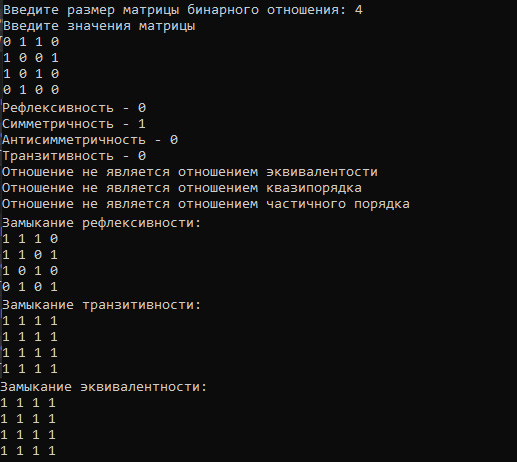
Построение замыкания эквивалентности – О(N + N2 + N4 + N2) ~ О(N4);

В суммепрограмма работает за O(N4).

3.2 Результаты тестирования программ







3.3 Код программы

#include "iostream"

using namespace std;

bool reflexivity(int\*\* mas, int N) {

for (int i = 0; i < N; i++)

if (mas[i][i] == 0)

return false;

return true;

}

bool symmetry(int\*\* mas, int N) {

for (int i = 0; i < N; i++)

for (int j = i; j < N; j++)

if (mas[i][j] != mas[j][i])

return false;

return true;

}

bool antisymmetry(int\*\* mas, int N) {

for (int i = 0; i < N; i++)

for (int j = i; j < N; j++)

if (i != j && mas[i][j] == mas[j][i])

return false;

return true;

}

bool transitivity(int\*\* mas, int N) {

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

for (int k = 0; k < N; k++)

if (mas[i][j] == 1 && mas[j][k] == 1 && mas[i][k] != 1)

return false;

return true;

}

void make\_reflexivity(int\*\* mas, int N, bool flag) {

for (int i = 0; i < N; i++)

mas[i][i] = 1;

if (flag) {

cout << "Замыкание рефлексивности:\n";

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++)

cout << mas[i][j] << " ";

cout << endl;

}

}

}

void make\_symmetry(int\*\* mas, int N, bool flag) {

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

if (mas[i][j] == 1)

mas[j][i] = 1;

if (flag) {

cout << "Замыкание симметричности:\n";

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++)

cout << mas[i][j] << " ";

cout << endl;

}

}

}

void make\_transitivity(int\*\* mas, int N, bool flag) {

for (int n = 0; n < N; n++)

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

for (int k = 0; k < N; k++)

if (mas[i][j] == 1 && mas[j][k] == 1)

mas[i][k] = 1;

if (flag) {

cout << "Замыкание транзитивности:\n";

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++)

cout << mas[i][j] << " ";

cout << endl;

}

}

}

void make\_equivalence(int\*\* mas, int N) {

make\_reflexivity(mas, N, false);

make\_symmetry(mas, N, false);

make\_transitivity(mas, N, false);

cout << "Замыкание эквивалентности:\n";

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++)

cout << mas[i][j] << " ";

cout << endl;

}

}

void checking(int\*\* mas, int N) {

int ref = reflexivity(mas, N), sym = symmetry(mas, N), antisym = antisymmetry(mas, N), tran = transitivity(mas, N);;

cout << "Рефлексивность - " << ref << endl << "Симметричность - " << sym << endl << "Антисимметричность - " << antisym << endl << "Транзитивность - " << tran << endl;

if (ref == 1 && sym == 1 && tran == 1)

cout << "Отношение является отношением эквивалентости\n";

else

cout << "Отношение не является отношением эквивалентости\n";

if (ref == 1 && tran == 1)

cout << "Отношение является отношением квазипорядка\n";

else

cout << "Отношение не является отношением квазипорядка\n";

if (ref == 1 && antisym == 1 && tran == 1)

cout << "Отношение является отношением частичного порядка\n";

else

cout << "Отношение не является отношением частичного порядка\n";

}

void text() {

cout << endl << "1 - Ввести новую матрциу\n2 - Проверить свойства\n3 - Построить замыкание рефлексивности\n4 - Построить замыкание симметричности\n";

cout << "5 - Построить замыкание транзитивности\n6 - Построить замыкание эквивалентности\n7 - Выход\n\n";

}

int main() {

setlocale(LC\_ALL, "ru");

int N;

cout << "Введите размер матрицы бинарного отношения: ";

cin >> N;

int\*\* matrix = new int\* [N];

for (int i = 0; i < N; i++)

matrix[i] = new int[N];

cout << "Введите значения матрицы\n";

int x;

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++) {

cin >> x;

matrix[i][j] = x;

}

for (;;) {

text();

cin >> x;

int\*\* newMatrix = new int\* [N];

for (int i = 0; i < N; i++) {

newMatrix[i] = new int[N];

for (int j = 0; j < N; j++)

newMatrix[i][j] = matrix[i][j];

}

switch (x) {

case 1:

cout << "Введите размер матрицы бинарного отношения: ";

cin >> N;

matrix = new int\* [N];

for (int i = 0; i < N; i++)

matrix[i] = new int[N];

cout << "Введите значения матрицы\n";

int x;

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++) {

cin >> x;

matrix[i][j] = x;

}

break;

case 2:

checking(matrix, N);

break;

case 3:

make\_reflexivity(newMatrix, N, true);

break;

case 4:

make\_symmetry(newMatrix, N, true);

break;

case 5:

make\_transitivity(newMatrix, N, true);

break;

case 6:

make\_equivalence(newMatrix, N);

break;

case 7:

return 0;

default:

cout << "Error. Try again";

}

}

}

ЗАКЛЮЧЕНИЕ

В ходе данной работы были изучены и реализованы алгоритмы проверки бинарных отношений на рефлексивность, симметричность, антисимметричность, транзитивность, эквивалентность, квазипорядок и частичный порядок, а также алгоритмы построения основных замыканий.