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 Алгоритм проверка бин. отношений на наличие свойств………..4

2.3 Отношение эквивалентности………………………………………5

2.4 Отношение квазипорядка………………………………………….6

2.5 Отношение частичного порядка…………………………………...6

2.6 Системы замыкания на множестве бинарных отношений……….7

2.7 Алгоритм построения основных замыканий бин. отношений…...8

3 Результаты работы…………………………………………………………..10

3.1 Оценки сложности рассмотренных алгоритмов………………...10

3.2 Результаты тестирования программ……………………………..10

3.3 Код программы……………………………………………………13

ЗАКЛЮЧЕНИЕ……………………………………………………………….16

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, *a,b* ∈ A;

– антисимметричным, если (a, b) ∈ ρ ^ (b, a) ∈ ρ ⇒ a = b, *a,b* ∈ A;

– транзитивным, если (a, b) ∈ ρ ^ (b, c) ∈ ρ ⇒ (a, c) ∈ ρ, *a,b,c* ∈ A.

Алгоритм проверки бинарного отношения на рефлексивность:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «1» или «0».

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1}. Вычислить *d* = mas[a][a]. Если *d* = 0, то ответ «0».

Шаг 2. Если «Шаг 1» не выдал ответ «0», то ответ «1» .

Алгоритм проверки бинарного отношения на симметричность:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «1» или «0».

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1} и для *a* выбрать *b* ∈ {0, 1, …, N - 1}. вычислить *d* = *mas*[a][b] и *v* = *mas*[b][a]. Если *d* не равно *v*, то ответ «0».

Шаг 2. Если «Шаг 1» не выдал ответ «0», то ответ «1».

Алгоритм проверки бинарного отношения на антисимметричность:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «1» или «0».

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1} и для *a* выбрать *b* ∈ {0, 1, …, N - 1}. вычислить *d* = *mas*[a][b] и *v* = *mas*[b][a]. Если *a* не равно *b* и *d* = *v*, то ответ «0».

Шаг 2. Если «Шаг 1» не выдал ответ «0», то ответ «1».

2.2 Алгоритм проверки бинарного отношения на транзитивность:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «1» или «0».

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1}, для *a* выбрать *b* ∈ {0, 1, …, N - 1} и для *b* выбрать *c* ∈ {0, 1, …, N - 1}. вычислить *d* = *mas*[a][b], *v* = *mas*[b][c] и *s* = *mas*[a][c]. Если *d* = 1, *v* = 1 и *s* не равно 1, то ответ «0».

Шаг 2. Если «Шаг 1» не выдал ответ «0», то ответ «1».

Существует три типа бинарных отношений:

– отношение эквивалентности

– отношение квазипорядка

– отношение частичного порядка

2.3 Отношение эквивалентности

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

Алгоритм проверки отношения на эквивалентность:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «Отношение является отношением эквивалентности» или «Отношение является отношением эквивалентности».

Шаг 1. Выполняется алгоритм проверка бин. отношения на рефлексивности и результат сохраняется в переменную *ref*. Выполняется алгоритм проверка бин. отношения на симметричность и результат сохраняется в переменную *sym*. Выполняется алгоритм проверка бин. отношения на транзитивность и результат сохраняется в переменную *tran*.

Шаг 2. Если *ref* = 1, *sym* = 1 и *tran* = 1, то ответ «Отношение является отношением эквивалентности». Если хотя бы одно условие не выполнено, то ответ «Отношение не является отношением эквивалентности».

2.4 Отношение квазипорядка

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

Алгоритм проверки отношения на квазипорядок:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «Отношение является отношением квазипорядка» или «Отношение является отношением квазипорядка».

Шаг 1. Выполняется алгоритм проверка бин. отношения на рефлексивности и результат сохраняется в переменную *ref*. Выполняется алгоритм проверка бин. отношения на транзитивность и результат сохраняется в переменную *tran*.

Шаг 2. Если *ref* = 1 и *tran* = 1, то ответ «Отношение является отношением квазипорядка». Если хотя бы одно условие не выполнено, то ответ «Отношение не является отношением квазипорядка».

2.5 Отношение частичного порядка

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

Алгоритм проверки отношения на частичный порядок:

*Вход.* Матрица бинарного отношения *mas* и размер матрицы *N*.

*Выход.* «Отношение является отношением частичного порядка» или «Отношение является отношением частичного порядка».

Шаг 1. Выполняется алгоритм проверка бин. отношения на рефлексивности и результат сохраняется в переменную *ref*. Выполняется алгоритм проверка бин. отношения на антисимметричность и результат сохраняется в переменную anti*sym*. Выполняется алгоритм проверка бин. отношения на транзитивность и результат сохраняется в переменную *tran*.

Шаг 2. Если *ref* = 1, anti*sym* = 1 и *tran* = 1, то ответ «Отношение является отношением частичного порядка». Если хотя бы одно условие не выполнено, то ответ «Отношение не является отношением частичного порядка».

2.6 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.*

2.7 Алгоритмы построения основных замыканий бинарных отношений

Алгоритм построения замыкания эквивалентности бинарного отношений

Вход. Матрица бинарного отношения mas и размер матрицы N.

Выход. Матрица бинарного отношения с замыканием эквивалентности.

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1}. Присвоить mas[a][a] = 1.

Шаг 2. Вывести матрицу бин. отношения с замыканием эквивалентности.

Алгоритм построения замыкания симметричности бинарного отношений

Вход. Матрица бинарного отношения mas и размер матрицы N.

Выход. Матрица бинарного отношения с замыканием симметричности.

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1} и для *a* выбрать *b* ∈ {0, 1, …, N - 1}. вычислить *d* = *mas*[a][b]. Если *d* = 1, то присвоить *mas*[b][a] = 1.

Шаг 2. Вывести матрицу бин. отношения с замыканием симметричности.

Алгоритм построения замыкания транзитивности бинарного отношений

Вход. Матрица бинарного отношения mas и размер матрицы N.

Выход. Матрица бинарного отношения с замыканием транзитивности.

Шаг 1. Выбрать *a* ∈ {0, 1, …, N - 1}, для *a* выбрать *b* ∈ {0, 1, …, N - 1} и для *b* выбрать *c* ∈ {0, 1, …, N - 1}. Вычислить *d* = *mas*[a][b] и *v* = *mas*[b][c]. Если *d* = 1 и *v* = 1, то присвоить *mas*[a][c] = 1.

Шаг 2. Повторить «Шаг 1» N раз, чтобы построить транзитивность для новых единиц в матрице.

Шаг 3. Вывести матрицу бин. отношения с замыканием симметричности.

Алгоритм построения замыкания эквивалентности бинарного отношений

Вход. Матрица бинарного отношения mas и размер матрицы N.

Выход. Матрица бинарного отношения с замыканием эквивалентности.

Шаг 1. Для матрицы бин. отношения применить алгоритм построения замыкания рефлексивности.

Шаг 2. Для матрицы бин. отношения с замыканием эквивалентности применить алгоритм построения замыкания симметричности.

Шаг 3. Для матрицы бин. отношения с замыканиями эквивалентности и симметричности применить алгоритм построения замыкания транзитивности.

Шаг 4. Вывести матрицу бин. отношения с замыканием симметричности

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);

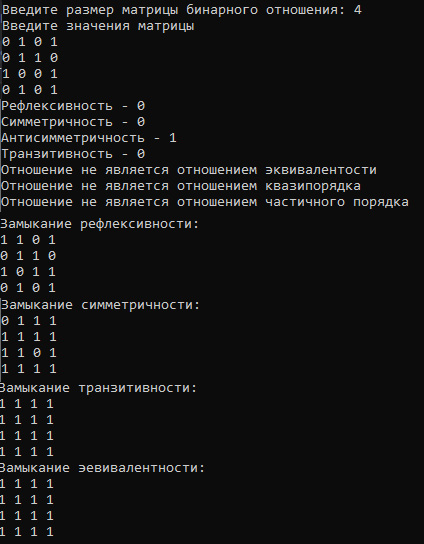
Построение замыкания симметричности – О(N2 + N2) ~ О(N2);

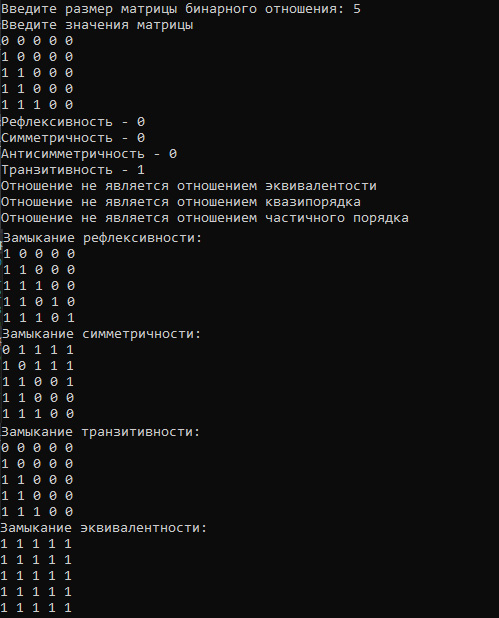
Построение замыкания транзитивности – О(N4 + N2) ~ О(N4);

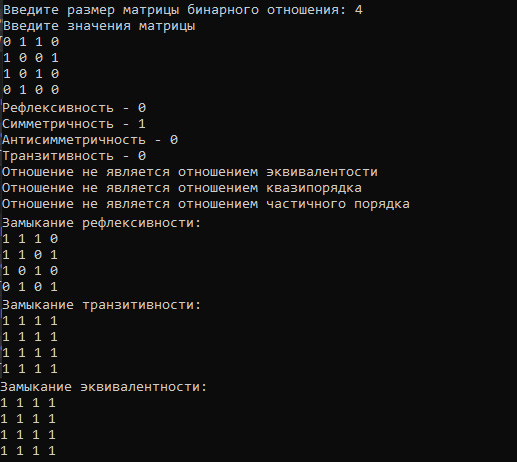
Построение замыкания эквивалентности – О(N + N2 + N4 + N2) ~ О(N4);

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.