Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Институт информационных технологий

Кафедра ИС

# ОТЧЕТ

по лабораторной работе №1

ИССЛЕДОВАНИЕ ПРИМЕНЕНИЯ АППАРАТА БИНАРНЫХ ОТНОШЕНИЙ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ВЫБОРА АЛЬТЕРНАТИВ

Выполнил:

ст. гр. ИС/б-21-2-о

Мовенко К. М.

Проверил:

Кротов К.В.

Севастополь

2024

# ЦЕЛЬ РАБОТЫ

Исследовать применение аппарата бинарных отношений при принятии решений по выбору альтернатив.

# ЗАДАНИЕ

Вариант 3: выполнить разработку программы, реализующей определение упорядоченного множества решений для множества , руководствуясь заданной формой графа отношений (Рисунок 1).

При разработке программы использовать правила формирования множества с учётом рассмотрения вершин-приемников на каждом шаге алгоритма. При формировании упорядоченного множества решений указывать номер яруса, на котором находятся решения.

Определить порядок решений аналитически. Выполнить сравнение полученных с использованием процедуры результатов с результатами, полученными аналитически.

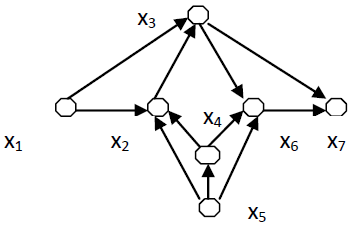


Рисунок 1 – Граф отношений предпочтения 1

Изменить в реализуемой программе исходные данные (Рисунок 2). Выполнить аналитическое построение множества для этих данных и сравнить его с результатами, полученными с использованием процедуры.

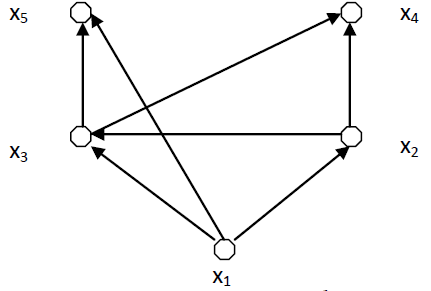


Рисунок 2 – Граф отношений предпочтения 2

# ХОД РАБОТЫ

Была построена матрица отношений для графа №1.

Была построена матрица отношений для графа №2.

* 1. Ручное формирование множества

Рассмотрим граф №1 и сформируем множества MaxR вручную.

1. Решения x1 и x5 не доминируемы и не эквивалентны с другими решениями. Это значит, что решения x1 и x5 в множество MaxR включаем.
2. Решение x4 доминируемо решением x5. Оно не эквивалентно ни с одним решением. Это значит, что x4 в множество MaxR не включаем.
3. Решение x2 доминируемо решениями x1, x4 и x5. Оно не эквивалентно ни с одним решением. Это значит, что x2 в множество MaxR не включаем.
4. Решение x3 доминируемо решениями x1 и x2. Оно не эквивалентно ни с одним решением. Это значит, что x3 в множество MaxR не включаем.
5. Решение x6 доминируемо решениями x3, x4 и x5. Оно не эквивалентно ни с одним решением. Это значит, что x6 в множество MaxR не включаем.
6. Решение x7 доминируемо решениями x3 и x6. Оно не эквивалентно ни с одним решением. Это значит, что x7 в множество MaxR не включаем.

В результате MaxR состоит из решений x1 и x5.

Рассмотрим граф №2 и сформируем множества MaxR вручную.

1. Решение x1 не доминируемо и не эквивалентно с другими решениями. Это значит, что решения x1 в множество MaxR включаем.
2. Решение x2 доминируемо решением x1. Оно не эквивалентно ни с одним решением. Это значит, что x2 в множество MaxR не включаем.
3. Решение x3 доминируемо решениями x1 и x2. Оно не эквивалентно ни с одним решением. Это значит, что x3 в множество MaxR не включаем.
4. Решение x4 доминируемо решениями x3 и x2. Оно не эквивалентно ни с одним решением. Это значит, что x4 в множество MaxR не включаем.
5. Решение x5 доминируемо решениями x1 и x3. Оно не эквивалентно ни с одним решением. Это значит, что x5 в множество MaxR не включаем.
   1. Определение множества решений.

Было определено упорядоченное множество решений для графа №1.

Таблица 1 – Упорядоченное множество эффективных решений для графа №1

|  |  |
| --- | --- |
| Ярус | Эффективное решение |
| 1 | x1, x5 |
| 2 | x4 |
| 3 | x2 |
| 4 | x3 |
| 5 | x6 |
| 6 | x7 |

Было определено упорядоченное множество решений для графа №2.

Таблица 2 – Упорядоченное множество эффективных решений для графа №2

|  |  |
| --- | --- |
| Ярус | Эффективное решение |
| 1 | x1 |
| 2 | x2 |
| 3 | x3 |
| 4 | x4, x5 |

Написана программа для определения множества решений MaxR для множества X на основе заданного графа с учетом рассмотрения вершин-приемников, для формирования упорядоченного множества решений с указанием номера яруса и определения эффективного решения

# ТЕКСТ ПРОГРАММЫ

Листинг 1 – Программный код

#include <iostream>

using namespace std;

const int n = 7; // число решений

// вывод матрицы отношений

void printA(int a[n][n])

{

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

{

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

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

cout << endl;

}

cout << endl;

}

int main()

{

int a[n][n] = {

{0, 1, 1, 0, 0, 0, 0},

{0, 0, 1, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 1, 1},

{0, 1, 0, 0, 0, 1, 0},

{0, 1, 0, 1, 0, 1, 0},

{0, 0, 0, 0, 0, 0, 1},

{0, 0, 0, 0, 0, 0, 0}

};

printA(a);

int MaxR[n]; // массив упорядоченных решений

int K = 0, K1 = 0; // количество элементов, добавленных в MaxR

int count = n; // счётчик

int layer = 1; // текущий ярус

while (count > 0)

{

cout << "Layer " << layer++ << ": ";

// определение исключаемых элементов

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

{

int sum = 0;

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

sum += a[i][j];

if (sum == 0)

{

cout << "x" << i + 1 << " ";

MaxR[K++] = i;

}

}

cout << endl;

// обнуление отношений с исключаемыми элементами

for (int q = K1; q < K; q++)

{

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

{

a[x][MaxR[q]] = 0;

}

}

// исключение элементов, добавленных в MaxR

for (int q = K1; q < K; q++)

{

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

{

a[MaxR[q]][x] = 1;

}

}

printA(a);

K1 = K;

count = n - K;

}

// вывод упорядоченных результатов

cout << "Multiple solutions: ";

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

cout << "x" << MaxR[i] + 1 << " ";

}

# ВЫВОД