Министерство науки и высшего образования Российской Федерации
Севастопольский государственный университет
Кафедра ИС

Отчет
по лабораторной работе №1
«Исследование применения аппарата бинарных отношений для решения задачи выбора альтернатив»
по дисциплине
«ТЕОРИЯ ПРИНЯТИЯ РЕШЕНИЙ»

Выполнил студент группы ИС/б-17-2-о
Горбенко К. Н.
Проверил
Кротов К. В.

Севастополь
2020

1 ЦЕЛЬ РАБОТЫ

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

2 ЗАДАНИЕ НА РАБОТУ

1. Для **Варианта 1** задания на работу, связанного с формированием подмножества максимальных элементов MaxR множества X, необходимо по заданному варианту графа отношений предпочтения между решениями сформировать матрицу A отношения R (где R – отношение $\succ$). При этом убедиться, что первый элемент множества X является строго независящим от других решений.

2. Выполнить формирование множества MaxR вручную для заданного вида графа и соответствующего ему вида матрицы A.

3. Выполнить формирование программного кода соответствующей процедуры определения множества MaxR, при этом возможно руководствоваться ориентировочным видом процедуры определения этого множества, предложенным в теоретическом введении данной лабораторной работы.

4. Выполнить вывод результатов работы процедуры и сравнить полученные в процедуре результаты с результатами, сформированными аналитически.

5. Изменить исходные данные программы, используя графы отношений из примера 5 (Рис 7). Проверить получаемые с использованием процедуры результаты с аналитическим результатами, формируемыми для этих графов.

3 ХОД РАБОТЫ

3.1 Матрица отношений

Сформируем матрицу отношения $\succ$ для графа отношений, соответствующего варианту № 1, изображенного на рисунке 1.

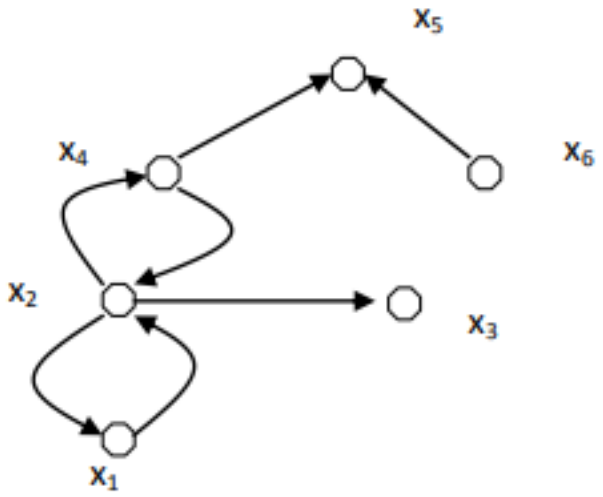

Рисунок 1 – Граф отношений варианта № 1

Матрица отношения выглядит следующим образом:

$$\begin{pmatrix} 0,1,0,0,0,0 \\ 1,0,1,1,0,0 \\ 0,0,0,0,0,0 \\ 0,1,0,0,1,0 \\ 0,0,0,0,0,0 \\ 0,0,0,0,1,0 \end{pmatrix}$$

3.2 Ручное формирование множества $Max_R X$

Рассмотрим каждую вершину графа, изображенного на рисунке 1 на предмет включения во множество $Max_R X$.

1. Решение x1 не является доминируемым, оно эквивалентно решениям x2 и x4, при этом ни одно решение не доминирует над ними. Из этого следует, что решения x1, x2 и x4 должны быть включены во множество $Max_R X$.

2. Решение x3 доминируемо решением x2, в множество $Max_R X$ не включаем.

3. Решение x5 доминируемо решениями x4 и x6, в множество $Max_R X$ не включаем.

4. Решение x6 никаким решением не доминируемо и ни с каким решением не эквивалентно, включим его в множество $Max_R X$.

Итоговое множество $Max_R X$ состоит из следующих элементов: x1, x2, x4 и x6.

3.3 Реализация процедуры формирования множества $Max_R X$

Реализованная процедура формирования множества $Max_R X$ приведена на следующем листинге (полный программный код приведен в приложении):

```
let private getMaxR (matrix: int list list): int list =
    let size = matrix |> List.length
    let maxR = [| for i in [0..size-1] do 1 |]

    for i in [0..size - 1] do
        for j in [0..size - 1] do
            if matrix.[i].[j] = 1 && matrix.[j].[i] = 0 then
                maxR.[j] <- 0
            if matrix.[j].[i] = 1 && maxR.[j] = 0 then
                maxR.[j] <- 0

    for i in [0..size - 1] do
        for j in [0..size - 1] do
            if matrix.[i].[j] = 1 && matrix.[j].[i] = 1 && maxR.[j] = 0 then
                maxR.[i] <- 0;

    maxR |> List.ofArray
```

Результат работы программы приведен на рисунке 2. Множество $Max_R X$, сформированное с помощью программы совпадает со сформированным ранее вручную.

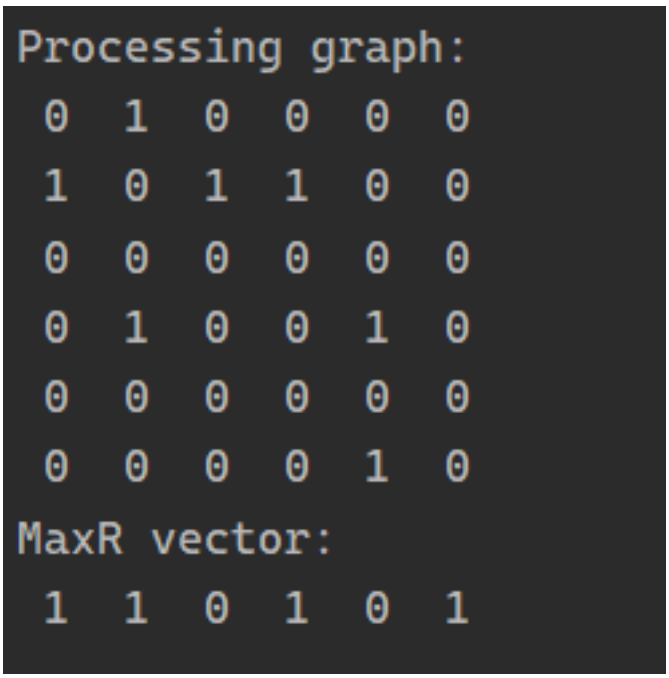

Рисунок 2 – Результат работы программы формирования множества максимальных решений

3.4 Проверка работы программы на графах рисунка 7

Включим во входные данные программы матрицы отношений графов с рисунка 7:

```
let picture7aRelationGraph = [
    [0; 0; 0; 1; 0]
    [0; 0; 1; 0; 0]
    [0; 0; 0; 1; 1]
    [0; 0; 1; 0; 0]
    [0; 0; 1; 0; 0]
]

let picture7bRelationGraph = [
    [0; 1; 0; 1; 0]
    [1; 0; 1; 0; 0]
    [0; 0; 0; 1; 1]
    [0; 0; 1; 0; 0]
    [0; 0; 0; 0; 0]
]
```

Результат работы программы для таких входных данных изображен на рисунке 3

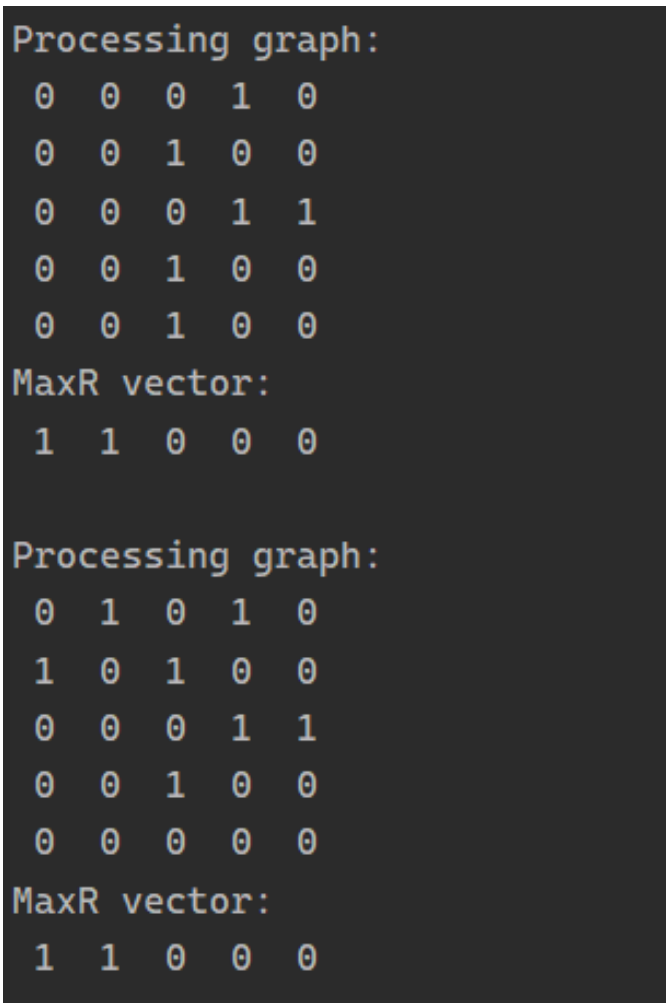

Рисунок 3 – Результат работы программы формирования множества максимальных решений для графов с рисунка 7

Все результаты совпадают с аналитическими.

ВЫВОДЫ

В ходе лабораторной работы была рассмотрена задача выбора оптимальных решений из альтернатив с помощью аппарата бинарных отношений. В частности, была рассмотрена ситуация, когда для списка альтернатив не может быть сформировано множество предпочитаемых решений.

Для решения задачи было сформировано множество $Max_R X$ максимальных решений по следующим правилам:

1. решение исключается из $Max_R X$ если оно доминируемо другим решением;

2. решение исключается из $Max_R X$ если оно эквивалентно другому решению, которое не включено в $Max_R X$;

3. если решение ником не доминируемо, то оно включается в $Max_R X$.

В результате максимальные множества, полученные аналитически и программно, совпали.

ПРИЛОЖЕНИЕ

Программный код приложения:

Файл Program.fs:

```
module BinaryRelations.Program

open System

let private printMatrix (matrix: int list list): unit =
    let size = matrix |> List.length
    for i in [0..size - 1] do
        for j in [0..size - 1] do
            matrix.[i].[j] |> printf " %d " |> ignore
        printfn "" |> ignore

let private printVector (vector: int list): unit =
    vector |> List.iter (fun x -> x |> printf " %d " |> ignore)

let private getMaxR (matrix: int list list): int list =
    let size = matrix |> List.length
    let maxR = [| for i in [0..size-1] do 1 |]

    for i in [0..size - 1] do
        for j in [0..size - 1] do
            if matrix.[i].[j] = 1 && matrix.[j].[i] = 0 then
                maxR.[j] <- 0
            if matrix.[j].[i] = 1 && maxR.[j] = 0 then
                maxR.[j] <- 0

    for i in [0..size - 1] do
        for j in [0..size - 1] do
            if matrix.[i].[j] = 1 && matrix.[j].[i] = 1 && maxR.[j] = 0 then
                maxR.[i] <- 0;

    maxR |> List.ofArray

[<EntryPoint>]
let main (_: string array): int =
    let matrices = [
        Data.relationGraphByVariant
        Data.picture5RelationGraph
        Data.picture7aRelationGraph
        Data.picture7bRelationGraph
        Data.relationGraphWithNoDominatingAlternative
    ]

    matrices
```

```
44      |> List.map (fun x -> x, getMaxR x)
45      |> List.iter (fun x ->
46              let matrix, maxR = x
47
48              printfn "%sProcessing graph:" (String.replicate 2 Environment.
                    NewLine) |> ignore
49              matrix |> printMatrix
50              printfn "MaxR vector:" |> ignore
51              matrix |> getMaxR |> printVector
52          )
53
54      0
```

Файл Data.fs:

```
 1 module BinaryRelations.Data
 2
 3 let relationGraphByVariant = [
 4      [0; 1; 0; 0; 0; 0]
 5      [1; 0; 1; 1; 0; 0]
 6      [0; 0; 0; 0; 0; 0]
 7      [0; 1; 0; 0; 1; 0]
 8      [0; 0; 0; 0; 0; 0]
 9      [0; 0; 0; 0; 1; 0]
10 ]
11
12 let picture5RelationGraph = [
13      [0; 0; 1; 0; 0]
14      [0; 0; 1; 0; 0]
15      [0; 1; 0; 1; 1]
16      [0; 0; 1; 0; 0]
17      [0; 0; 0; 0; 0]
18 ]
19
20 let picture7aRelationGraph = [
21      [0; 0; 0; 1; 0]
22      [0; 0; 1; 0; 0]
23      [0; 0; 0; 1; 1]
24      [0; 0; 1; 0; 0]
25      [0; 0; 1; 0; 0]
26 ]
27
28 let picture7bRelationGraph = [
29      [0; 1; 0; 1; 0]
30      [1; 0; 1; 0; 0]
31      [0; 0; 0; 1; 1]
32      [0; 0; 1; 0; 0]
33      [0; 0; 0; 0; 0]
34 ]
```

```
35
36 let relationGraphWithNoDominatingAlternative = [
37     [0; 1; 0; 0; 0; 0; 0]
38     [1; 0; 1; 0; 0; 0; 0]
39     [0; 0; 0; 1; 0; 0; 0]
40     [0; 0; 1; 0; 1; 1; 0]
41     [0; 0; 0; 0; 0; 0; 0]
42     [1; 0; 0; 1; 0; 0; 1]
43     [0; 0; 0; 0; 0; 0; 0]
44 ]
```