Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рязанский государственный радиотехнический университет

имени В.Ф. Уткина»

Кафедра АСУ

Отчёт о практической работе №2

«Реализация алгоритма Парето»

По дисциплине

«Математические основы принятия решений»

Выполнил ст. гр. 135

Бардин М.С.

Проверил:

Челебаева Ю.А

Челебаев С.В.

Рязань 2023

**Цель работы**

Реализовать алгоритм Парето на языке C#.

**Алгоритм Парето**

**Шаг 1.** Положить , , . Тем самым образуется так называемое **текущее множество парето-оптимальных векторов**, которое в начале работы алгоритма совпадает с множеством , а в конце − составит искомое множество парето-оптимальных векторов. Алгоритм устроен таким образом, что искомое множество парето-оптимальных векторов получается из *Y* последовательным удалением заведомо неоптимальных векторов.

**Шаг 2.** Проверить выполнение неравенства . Если оно оказалось истинным, то перейти к Шагу 3. В противном случае перейти к Шагу 5.

**Шаг 3.** Удалить из текущего множества векторов  вектор , так как он не является парето-оптимальным. Затем перейти к Шагу 4.

**Шаг 4.** Проверить выполнение неравенства . Если оно имеет место, то положить  и вернуться к Шагу 2. В противном случае – перейти к Шагу 7.

**Шаг 5.** Проверить справедливость неравенства . В том случае, когда оно является истинным, перейти к Шагу 6. В противном случае – вернуться к Шагу 4.

**Шаг 6.** Удалить из текущего множества векторов  вектор  и перейти к Шагу 7.

**Шаг 7.** Проверить выполнение неравенства . В случае истинности этого неравенства следует последовательно положить , а затем . После этого необходимо вернуться к Шагу 2. В противном случае (т.е. когда ) вычисления закончить. К этому моменту множество парето-оптимальных векторов построено полностью.

**Исходные данные (Вариант 2)**

|  |  |  |
| --- | --- | --- |
| № Варианта | Количество элементов в множестве Парето | Количество критериев |
| 2 | 6 | 7 |

**Практическая часть**

using System;

using System.Drawing;

public class Program

{

public static void ShowArray(int[,] myArr) // Отображение массива в таблице

{

for (int i = 0; i < 6; i++)

{

Console.WriteLine();

for (int j = 0; j < 7; j++)

{

Console.Write(myArr[i, j] + " ");

}

}

}

public static void GenArray(int[,] myArr) // Генерация массива случайных значений

{

Random rand = new Random();

int k = 0;

int t;

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 7; j++)

{

t = rand.Next(1, 10);

if (t > k)

myArr[i, j] = t - k;

else myArr[i, j] = 0;

}

k = k + 1;

}

}

public static bool CompareArray(int[] arr1, int[] arr2) // Сравнение одномерных массивов

{

int count = 0;

for (int n = 0; n < arr1.Length; n++)

{

if (arr1[n] >= arr2[n])

count += 1;

}

if (count == arr1.Length)

return true;

else return false;

}

public static void FillArray(int[,] myArr, int[] arr3, int size, int arr\_row) // Заполнение одномерного массива для сравнения

{

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

{

arr3[i] = myArr[arr\_row, i];

}

}

public static void ZeroArray(int[,] myArr, int size\_row, int arr\_row) // Заполнение одномерного массива в (удаление массива)

{

for (int i = 0; i < size\_row; i++)

{

myArr[arr\_row, i] = 0;

}

}

public static void BestResult(int[,] myArr) // Вывод лучших результатов

{

for (int i = 0; i < 6; i++)

{

if (myArr[i, 0] != 0)

{

if (i == 6)

Console.WriteLine("Сигнал N " + Convert.ToString(i + 1));

else Console.WriteLine("Сигнал N " + Convert.ToString(i + 1) + "\n");

}

}

}

public static void Main()

{

int[,] myArr = new int[6, 7]; // Текущее множество парето-оптимальных векторов

GenArray(myArr);

Console.WriteLine("Исходный массив:");

ShowArray(myArr);

// Шаг 1

int i = 0;

int j = 1;

int N = 6;

Boolean flag = true;

do

{

int[] a = new int[7];

int[] b = new int[7];

FillArray(myArr, a, 7, i);

FillArray(myArr, b, 7, j);

if (CompareArray(a, b)) // Шаг 2

{

ZeroArray(myArr, 7, j); //Шаг 3

if (j < N) // Шаг 4

{

j = j + 1;

}

else

{

if (i < (N - 1)) // Шаг 7

{

i = i + 1;

j = i + 1;

}

else flag = false; // Конец вычислений

}

}

else

{

int[] c = new int[7];

int[] d = new int[7];

FillArray(myArr, c, 7, j);

FillArray(myArr, d, 7, i);

if (CompareArray(c, d)) // Шаг 5

{

ZeroArray(myArr, 7, i); // Шаг 6

if (i < (N - 1)) // Шаг 7

{

i = i + 1;

j = i + 1;

}

else flag = false; // Конец вычислений

}

else

{

if (j < N) // Шаг 4

{

j = j + 1;

}

else

{

if (i < (N - 1)) // Шаг 7

{

i = 1 + 1;

j = i + 1;

}

else flag = false; // Конец вычислений

}

}

}

} // конец do

while (flag);

Console.WriteLine();

Console.WriteLine("Результирующий массив:");

ShowArray(myArr);

Console.WriteLine();

BestResult(myArr);

}

}

Практическая часть

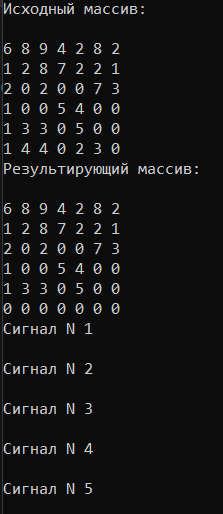


Рисунок 2 - Результат