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 №1

«Реализация симплекс-метода»

По дисциплине

«Математические основы принятия решений»

Выполнил ст. гр. 135

Бардин М.С

Проверил:

Челебаев С.В.

Челебаева Ю.А.

Рязань 2023

**Цель работы**

Реализация симплекс-метода на C#

**Краткие теоретические сведения**

Симплекс-метод является универсальным методом решения задач линейного программирования с любым числом переменных и с любым числом ограничений.

Исходная форма задачи, к которой непосредственно применим симплекс-метод, должна иметь специальный вид. Эта форма является частным случаем основной формы задач линейного программирования. Здесь также система ограничений представлена ограничениями-равенствами (линейными уравнениями) и условиями не отрицательности. Однако в равенствах, кроме того, выделяются так называемые **базисные переменные**. В каждом из равенств присутствует одна определенная базисная переменная, взятая с единичным коэффициентом, а в других равенствах ее нет. Число базисных переменных, таким образом, совпадает с числом ограничений-равенств в системе и обычно строго меньше общего числа переменных. Остальные переменные называются небазисными или свободными. Еще одно требование заключается в выполнении условия не отрицательности свободных членов *bi* в равенствах. Целевая функция задачи должна быть выражена только через небазисные переменные. Некоторые авторы называют такую форму представления задачи линейного программирования **канонической**.

Во многих случаях каноническая форма задачи получается автоматически при переходе от стандартной формы к основной с помощью введения новых переменных. Для этого требуется, чтобы свободные члены в неравенствах были неотрицательными, и все неравенства в (1.4) имели единственный знак «=».

**Практическая часть**

Вариант задания 2 (рисунок 1)

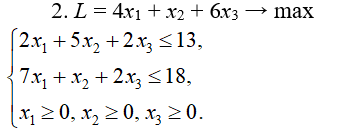


Рисунок 1- Вариант задания

**Листинг**

using System**;**using System**.**Collections**.**Generic**;**namespace Ex**.**\_1  
**{** public class Simplex  
 **{** private double**[,]** table**;** private int m**,** n**;** private List<int> basis**;** private Simplex**(**double**[,]** source**)  
 {** m = source**.**GetLength**(**0**);** n = source**.**GetLength**(**1**);** table = new double**[**m**,** n + m - 1**];** basis = new List<int>**();** for **(**var i = 0**;** i < m**;** i++**)  
 {** for **(**var j = 0**;** j < table**.**GetLength**(**1**);** j++**)  
 {** if **(**j < n**)** table**[**i**,** j**]** = source**[**i**,** j**];** else table**[**i**,** j**]** = 0**;  
 }** if **((**n + i**)** < table**.**GetLength**(**1**))  
 {** table**[**i**,** n + 1**]** = 1**;** basis**.**Add**(**n + i**);  
 }  
 }** n = table**.**GetLength**(**1**);  
 }** private double**[,]** Calculate**(**double**[]** result**)  
 {** while **(**!IsItEnd**())  
 {** var mainCol = FindMainCol**();** var mainRow = FindMainRow**(**mainCol**);** basis**[**mainRow**]** = mainCol**;** double**[,]** newTable = new double**[**m**,** n**];** for **(**int j = 0**;** j < n**;** j++**)** newTable**[**mainRow**,** j**]** = table**[**mainRow**,** j**]** / table**[**mainRow**,** mainCol**];** for **(**int i = 0**;** i < m**;** i++**)  
 {** if **(**i == mainRow**)** continue**;** for **(**int j = 0**;** j < n**;** j++**)** newTable**[**i**,** j**]** = table**[**i**,** j**]** - table**[**i**,** mainCol**]** \* newTable**[**mainRow**,** j**];  
 }** table = newTable**;  
 }** for **(**var i = 0**;** i < result**.**Length**;** i++**)  
 {** var k = basis**.**IndexOf**(**i + 1**);** if **(**k != -1**)** result**[**i**]** = table**[**k**,** 0**];** else  
 result**[**i**]** = 0**;  
 }** return table**;  
 }** private bool IsItEnd**()  
 {** bool flag = true**;** for **(**int j = 1**;** j < n**;** j++**)  
 {** if **(**table**[**m - 1**,** j**]** < 0**)  
 {** flag = false**;** break**;  
 }  
 }** return flag**;  
 }** private int FindMainCol**()  
 {** int mainCol = 1**;** for **(**int j = 2**;** j < n**;** j++**)** if **(**table**[**m - 1**,** j**]** < table**[**m - 1**,** mainCol**])** mainCol = j**;** return mainCol**;  
 }** private int FindMainRow**(**int mainCol**)  
 {** int mainRow = 0**;** for **(**int i = 0**;** i < m - 1**;** i++**)** if **(**table**[**i**,** mainCol**]** > 0**)  
 {** mainRow = i**;** break**;  
 }** for **(**int i = mainRow + 1**;** i < m - 1**;** i++**)** if **((**table**[**i**,** mainCol**]** > 0**)** &&  
 **((**table**[**i**,** 0**]** / table**[**i**,** mainCol**])** < **(**table**[**mainRow**,** 0**]** / table**[**mainRow**,** mainCol**])))** mainRow = i**;** return mainRow**;  
 }** class Program  
 **{** static void Main**()  
 {** double**[,]** table =  
 **{  
 {**13**,** 2**,** 5**,** 2**},  
 {**18**,** 7**,** 1**,** 2**},  
 {**0**,** -4**,** -1**,** -6**}  
 };** double**[]** result = new double**[**3**];** Simplex s = new Simplex**(**table**);** double**[,]** tableResult = s**.**Calculate**(**result**);** Console**.**WriteLine**(**"Решенная симплекс-таблица:"**);** for **(**int i = 0**;** i < tableResult**.**GetLength**(**0**);** i++**)  
 {** for **(**int j = 0**;** j < tableResult**.**GetLength**(**1**);** j++**)** Console**.**Write**(**tableResult**[**i**,** j**]** + " "**);** Console**.**WriteLine**();  
 }** Console**.**WriteLine**();** Console**.**WriteLine**(**"Решение:"**);** Console**.**WriteLine**(**"X[1] = " + result**[**0**]);** Console**.**WriteLine**(**"X[2] = " + result**[**1**]);** Console**.**WriteLine**(**"X[3] = " + result**[**2**]);** Console**.**ReadLine**();  
 }  
  
 }  
 }  
}**

**Результат алгоритма**

Результат выполнения симплекс-метода представлен на рисунке 2.

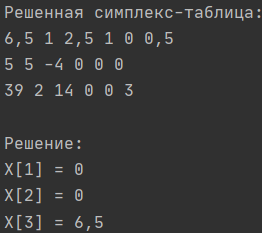


Рисунок 2- Вывод алгоритма

Заключение

Были приобретены навыки по реализации симплекс-метода.