**Міністерство освіти та науки України**

**Національний Авіаційний університет**

****

**Лабораторна робота 2.1**

з дисципліни «Інтелектуальні системи»

Виконав: студент групи ПІ-322

Царук С.О.

Прийняв: викладач

Клюєв Є. І.

Київ 2021

**Мета лабораторної роботи –** практично познайомитися з математичним апаратом, що застосовується в теорії ігор для обгрунтування прийняття стратегічних управлінських рішень. Навчитися використовувати метод мінімаксної і змішаної стратегії шляхом залучення аналітичних методів розв’язання гри для максимізації свого виграшу або мінімізації свого програшу в практиці управління. Показати можливість програмування типових задач теорії ігор в середовищі MS Visual Studio.

**Завдання**

1. Розв’язати типову задачу теорії ігор і підготувати звіт, який демонструє основні кроки пошуку найкращого розв’язку в платіжній матриці 3 × 3 за допомогою аналітичного методу мінімакса. Розробити алгоритм, що відображає основні кроки визначення оптимальної стратегії для двох гравців і знаходження нижньої та верхньої ціни гри на основі застосування принципу мінімакса і максиміна.
2. Підготувати звіт, що демонструє основні кроки пошуку найкращого рішення в платіжній матриці 3 × 3 не має сідлової точки, використовуючи один з методів алгебри (метод Гаусса, Зейделя, симплексний метод). Розробити програму, яка забезпечить визначення оптимальної стратегії для двох гравців і знаходження нижньої та верхньої ціни гри на основі застосування одного з методів лінійного програмування.

**Хід роботи**

1. **Постановка задачі та практичне розв’язання.**

Нехай є два гравця: гравець А і гравець B. Метою гравця А в грі є отримання максимальної роботоздатності програми, а метою гравця В – отримання мінімальної роботоздатності програми.

Метою гравця А в грі є отримання максимального виграшу, а метою гравця В – отримання максимального програшу. Під час гри гравець А може обрати одну з можливих стратегій: А1, А2 або А3, а гравець В відповісти стратегіями В1, В2 або В3. Метою гри є знаходження оптимальної стратегії для двох гравців, знаходження нижньої та верхньої ціни гри на основі застосування принципу мінімакса і максиміна.

Задано наступну платіжну матрицю:

|  |  |  |  |
| --- | --- | --- | --- |
| В  А | В1 | В2 | В3 |
| А1 | 8 | 5 | 7 |
| А2 | 3 | 3 | 6 |
| А3 | 4 | 6 | 1 |

Гравець А, обираючи стратегію Аі, розраховує на те, що гравець В відповість такою стратегією Вj, щоб виграш А був мінімальним. Тому спочатку необхідно знайти мінімальне з чисел в i-й рядку і позначимо його :

Випишемо числа (мінімуми рядків) поруч з матрицею праворуч у вигляді додаткового стовпця.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| В  А | В1 | В2 | В3 | min |
| А1 | 8 | 5 | 7 | 5 |
| А2 | 3 | 3 | 6 | 3 |
| А3 | 4 | 6 | 1 | 1 |

Вибираючи якусь стратегію , розраховуємо на те, що в результаті дії противника виграємо тільки . Діючи без жодного ризику вибираємо стратегію, для якої число максимально. Позначимо це максимальне значення a:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| В  А | В1 | В2 | В3 | min | max min |
| А1 | 8 | 5 | 7 | 5 | + |
| А2 | 3 | 3 | 6 | 3 | - |
| А3 | 4 | 6 | 1 | 1 | - |

Величина a називається ***нижньою ціною гри***, інакше – максимальним гарантованим виграшем виграшем або Максиміном. Та стратегія гравця А, яка відповідає Максиміну *a* називається максимінною стратегією.

З іншого боку, гравець В, обираючи стратегію Вj, розуміє що гравець А відповість такою стратегією Аі, щоб максимізувати свій виграш. Гравець В зацікавлений в тому, щоб звернути виграш противника А до мінімуму і для цього він переглядає всі свої стратегії, виділяючи для кожної з них максимальне значення виграшу. Виписуємо внизу матриці максимальні значення по стовпцях:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| В  А | В1 | В2 | В3 | min | max min |
| А1 | 8 | 5 | 7 | 5 | 5 |
| А2 | 3 | 3 | 6 | 3 |  |
| А3 | 4 | 6 | 1 | 1 |  |
| max | 8 | 6 | 7 |  |  |

і знайдемо з них мінімальне:

або

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| В  А | В1 | В2 | В3 | min | max min |
| А1 | 8 | 5 | 7 | 5 | 5 |
| А2 | 3 | 3 | 6 | 3 |  |
| А3 | 4 | 6 | 1 | 1 |  |
| max | 8 | 6 | 7 |  |  |
| min max |  | 6 |  |  |  |

Величина β називається ***верхньою ціною гри***, мінімаксним виграшем або МініМакс.

Відповідна виграшу β стратегія противника називається його мінімаксної стратегією. Дотримуючись своєї найбільш обережною мінімаксної стратегії, противник гарантований, що в кожному разі він програє не більш β.

1. **Розв’язання матричної гри в змішаних стратегіях.**

Задана гра не має сідлової точки (a < β), а отже ціна гри знаходиться в проміжку 3 ≤ y ≤ 5. Нехай SA = (p1, p2, p3) – оптимальна змішана стратегія гравця А, в якій стратегії A1, A2, A3 застосовуються з імовірностями p1, p2, ..., pm, причому p1 + p2 + p3 = 1.

Аналогічне позначення для змішаної стратегії противника буде:

SB = (q1, q2, q3), де q1 + q2 + q3 = 1.

Знайдемо оптимальну змішану стратегію для гравця А:

Маємо наступну систему рівнянь:

Перенесемо змінну у вліво від знаку рівності та вирішимо систему рівнянь методом Гаусса.

Запишемо систему в вигляді розширеної матриці:

Для зручності обчислення поміняємо рядки місцями:

Помножимо 3-й рядок на -1/3 та додамо до 4-го:

Помножимо 2-й рядок на -3/4 та додамо до 3-го:

Помножимо 1-й рядок на -1/2 та додамо до 2-го:

Помножимо 2-й рядок на 3/7 та додамо до 3-го:

Помножимо 3-й рядок на 28/117 та додамо до 4-го:

Помножимо рядки на відповідні множники, щоб отримати значення 1 на головній діагоналі:

Тепер початкову систему можна переписати в наступному вигляді:

Отже:

Отже, оптимальна змішана стратегія для гравця А: SA = (|-1/2|, 1, 1/2).

Знайдемо оптимальну змішану стратегію гравця В:

Маємо наступну систему рівнянь:

Запишемо систему в вигляді розширеної матриці:

Для зручності обчислення поміняємо рядки місцями:

Помножимо 3-й рядок на -1/3 та додамо до 4-го:

Помножимо 2-й рядок на -3/4 та додамо до 3-го:

Помножимо 1-й рядок на -1/2 та додамо до 2-го:

Помножимо 2-й рядок на 3/7 та додамо до 3-го:

Помножимо 3-й рядок на 28/117 та додамо до 4-го:

Помножимо рядки на відповідні множники, щоб отримати значення 1 на головній діагоналі:

Тепер початкову систему можна переписати в наступному вигляді:

Отже:

Отже, оптимальна змішана стратегія для гравця В: SВ = (|-1/2|, 1, 1/2).

Таким чином, при грі гравців А і В, мінімальною роботоздатністю програми буде характеристика, що знаходиться на перетині А3 та В3, та дорівнює 1, а максимальною роботоздатністю - на перетині А2 та В2, та дорівнює 3. Очікувана робота програми – з характеристикою роботоздатності 9/2.

Використання методу Мінімакса дозволяє на ранній стадії розробки проекта вирішити проблему знаходження стратегії, яка забезпечує отримання максимального прибутку гравцем A та мінімальних витрат гравцем B

Рішення заданої системи рівнянь дозволяє розрахувати вірогіднісні показники стратегій, які обгрунтовують вибір оптимальної стратегії розробки ПЗ для гравців A та B. Побудуємо матрицю вірогідностей для стратегій, використовуючи отримані значення. Знайдемо мінімальні значення за рядками та максимальні за стовпцями:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| В  А | В1 | В2 | В3 | min | max min |
| А1 | 1/4 | 1/2 | 1/4 | 1/2 |  |
| А2 | 1/2 | 1 | 1/2 | 1/2 | 1/2 |
| А3 | 1/4 | 1/2 | 1/4 | 1/2 |  |
| max | 1/4 | 1 | 1/4 |  |  |
| min max |  |  | 1/4 |  |  |

Знайдемо максимальний елемент α (мінімакс), що пов’язаний з оптимальною стратегією гравця B та мінімальний елемент β (максимін), що пов’язаний з оптимальною стратегією гравця A. В даній умові вірогіднісний показник стратегій гравця A дорівнює 1/2 та 1/4 для гравця В. Отже, можна зробити висновок, що оптимальною для гравця А буде будь-яка з трьох стратегій. Дя гравця В – стратегія В1 чи В3.

1. **Код програми.**

**MixedSolver.cs**

using System;

using System.Linq;

namespace IntSys\_Lab2.\_1.Model

{

public class MixedSolver

{

public double[,] Value { get; set; }

public double[] P { get; set; }

public double[] Q { get; set; }

public double Y { get; set; }

public bool CanSolve { get; set; }

public MixedSolver(double[,] value)

{

Value = value;

CanSolve = false;

Solve();

}

public void Solve()

{

double[] p = new double[Value.GetLength(0) + 1];

double[] q = new double[Value.GetLength(1) + 1];

int info;

alglib.densesolverreport rep;

double[,] pTmpA = new double[Value.GetLength(0) + 1, Value.GetLength(1) + 1];

Value.CopyTo(pTmpA);

for (int i = 0; i < pTmpA.GetLength(0) - 1; i++)

pTmpA[i, pTmpA.GetLength(1) - 1] = -1;

for (int i = 0; i < pTmpA.GetLength(1) - 1; i++)

pTmpA[pTmpA.GetLength(0) - 1, i] = 1;

double[] pTmpB = new double[Value.GetLength(0) + 1];

pTmpB[pTmpB.Length - 1] = 1;

alglib.rmatrixsolve(pTmpA, pTmpA.GetLength(0), pTmpB, out info, out rep, out p);

double[,] qTmpA = new double[Value.GetLength(0) + 1, Value.GetLength(1) + 1];

double[,] tmp = Value.Clone() as double[,];

tmp.Inverse();

tmp.CopyTo(qTmpA);

for (int i = 0; i < qTmpA.GetLength(0) - 1; i++)

qTmpA[i, qTmpA.GetLength(1) - 1] = -1;

for (int i = 0; i < qTmpA.GetLength(1) - 1; i++)

qTmpA[qTmpA.GetLength(0) - 1, i] = 1;

double[] qTmpB = new double[Value.GetLength(0) + 1];

qTmpB[qTmpB.Length - 1] = 1;

alglib.rmatrixsolve(qTmpA, qTmpA.GetLength(0), qTmpB, out info, out rep, out q);

Q = new double[q.Length - 1];

P = new double[p.Length - 1];

Array.Copy(p, Q, q.Length - 1);

Array.Copy(q, P, p.Length - 1);

if (q.Last() == p.Last())

{

CanSolve = true;

Y = q.Last();

}

}

}

}

**MiniMaxSolver.cs**

using System.Linq;

namespace IntSys\_Lab2.\_1.Model

{

public class MiniMaxSolver

{

public double[,] Values { get; set; }

public double[] MinA { get; set; }

public double[] MaxB { get; set; }

public double MaxA { get; set; }

public double MinB { get; set; }

public bool CanSolve { get; set; }

public MiniMaxSolver(double[,] values)

{

Values = values;

CanSolve = false;

Solve();

}

public void Solve()

{

MinA = new double[Values.GetLength(0)];

MaxB = new double[Values.GetLength(1)];

for (int i = 0; i < MinA.Length; i++)

MinA[i] = Values.GetRow(i).Min();

for (int i = 0; i < MaxB.Length; i++)

MaxB[i] = Values.GetCol(i).Max();

MaxA = MinA.Max();

MinB = MaxB.Min();

CanSolve = (MinB == MaxA);

}

}

}

**DoubleArrayExt.cs**

namespace IntSys\_Lab2.\_1.Model

{

public static class DoubleArrayExt

{

public static void Inverse(this double[,] arr)

{

for (int i = 0; i < arr.GetLength(0); i++)

for (int j = i; j < arr.GetLength(1); j++)

{

double tmp = arr[i, j];

arr[i, j] = arr[j, i];

arr[j, i] = tmp;

}

}

public static double[] GetRow(this double[,] arr, int j)

{

double[] ans = new double[arr.GetLength(1)];

for (int i = 0; i < arr.GetLength(1); i++)

ans[i] = arr[j, i];

return ans;

}

public static double[] GetCol(this double[,] arr, int j)

{

double[] ans = new double[arr.GetLength(0)];

for (int i = 0; i < arr.GetLength(0); i++)

ans[i] = arr[i, j];

return ans;

}

public static void CopyTo(this double[,] arrOrigin, double [,] arrCopy)

{

for(int i = 0; i < arrOrigin.GetLength(0); i++)

{

for(int j = 0; j < arrOrigin.GetLength(1); j++)

{

arrCopy[i, j] = arrOrigin[i, j];

}

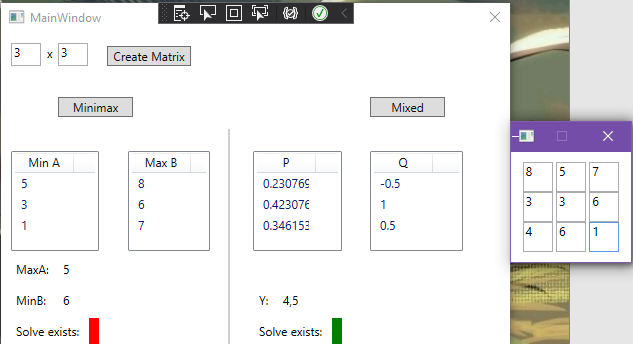
}

}

}

}

1. **Скріншоти.**



**Висновки**

Під час виконання лабораторної роботи я практично познайомилася з математичним апаратом, що застосовується в теорії ігор для обгрунтування прийняття стратегічних управлінських рішень. Навчилася використовувати метод мінімаксної і змішаної стратегії шляхом залучення аналітичних методів розв’язання гри для максимізації свого виграшу або мінімізації свого програшу в практиці управління. Показала можливість програмування типових задач теорії ігор в середовищі MS Visual Studio.