КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп’ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Лабораторна робота №2

з предмету «Математичні основи захисту інформації»

Варіант №3

Підготував:
Грищенко Юрій, ІПС-42

Київ – 2022

Теорія

Нехай I — деяка скінченна чи злічена множина, елементи якої називаються станами. Нехай деякий процес в момент часу n (де n=0,1,2,3…) може перебувати в одному із цих станів, а в час n+1 перейти в деякий інший стан (чи залишитися в тому ж). Кожен такий перехід називається **кроком**. Кожен крок не є точно визначеним. З певними ймовірностями процес може перейти в один з кількох чи навіть усіх станів.

Якщо імовірності переходу залежать лише від часу n і стану в якому перебуває процес в цей час і не залежать від станів в яких процес перебував у моменти 0, 1, … , n-1 то такий процес називається **дискретним ланцюгом Маркова**. Ланцюг Маркова повністю задається визначенням ймовірностей p_i перебування процесу в стані i ∈ I в час n=0 і ймовірностей $p_{ij}(n)$ переходу зі стану i ∈ I в стан j ∈ I в час n. Якщо ймовірності переходу не залежать від часу (тобто $p_{ij}(n)$ однакові для всіх n) то такий ланцюг Маркова називається **однорідним**. Саме однорідні ланцюги Маркова є найважливішими на практиці і найкраще вивченими теоретично

.

Послідовність дискретних випадкових величин $\{X_n\}_{n>=0}$ називається ланцюгом Маркова (з дискретним часом), якщо

$$\mathbb{P}(X_{n+1} = i_{n+1} \mid X_n = i_n, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = \mathbb{P}(X_{n+1} = i_{n+1} \mid X_n = i_n)$$

Тобто майбутні значення послідовності залежать лише від теперішнього стану і не залежать від минулих. Матриця P(n), де

$$P_{ij}(n) \equiv \mathbb{P}(X_{n+1} = j \mid X_n = i)$$

називається **матрицею ймовірностей переходу** на n-му кроці, а вектор $\mathbf{p} = (p_1, p_2, \ldots)^\top$ де $p_i \equiv \mathbb{P}(X_0 = i)$

називається **початковим розподілом** ланцюга Маркова.

Матриця ймовірностей переходу є стохастичною.
Якщо ланцюг Маркова однорідний, то $P_{ij}(n) = P_{ij}$

Варіант 3

Однорідний ланцюг Маркова задано матрицею P переходів з попереднього стану до наступного, а також вектор p_0 початкового розподілу ймовірностей. Знайти вектори розподілу ймовірностей p_3 та p_4. Обчислення виконати з точністю до 10^{-5}.

$$P = \begin{vmatrix} 0{,}1 & 0{,}3 & 0{,}6 \\ 0{,}2 & 0{,}5 & 0{,}3 \\ 0{,}6 & 0{,}2 & 0{,}2 \end{vmatrix}$$

p_0 = (0,2; 0,4; 0,4)

<u>Програма Java</u>

```java
package ua.knu.yurihr.infosec;

import java.util.Arrays;
```

```java
public class Main {
    private static class DoubleMatrix {
        private final double[][] rows;

        public DoubleMatrix(double[][] rows) {
            this.rows = rows;
        }

        public DoubleMatrix multiply(DoubleMatrix matrix2) {
            double[][] resultRows = new double[rows.length][];
            int resultWidth = matrix2.rows[0].length;

            for (int j = 0; j < rows.length; j++) {
                resultRows[j] = new double[resultWidth];

                for (int i = 0; i < resultWidth; i++) {
                    for (int k = 0; k < rows[j].length; k++) {
                        resultRows[j][i] += this.rows[j][k] * matrix2.rows[k][i];
                    }
                }
            }
            return new DoubleMatrix(resultRows);
        }

        public double[] multiplyByLeft(double[] vector) {
            double[] resultColumn = new double[rows.length];

            for (int i = 0; i < rows.length; i++) {
                for (int j = 0; j < rows.length; j++) {
                    resultColumn[i] += vector[j] * rows[j][i];
                }
            }
            return resultColumn;
        }

        @Override
        public String toString() {
            StringBuilder sb = new StringBuilder();
            for (int i = 0; i < rows.length; i++) {
                if (i == 0)
                    sb.append('[');
                else
                    sb.append(' ');
                sb.append(Arrays.toString(rows[i]));
                if (i == rows.length - 1)
                    sb.append(']');
                else
                    sb.append('\n');
            }
            return sb.toString();
        }
    }

    public static void main(String[] args) {
        double[] p0 = new double[]{0.2, 0.4, 0.4};
        System.out.println(Arrays.toString(p0));

        DoubleMatrix pMatrix = new DoubleMatrix(new double[][]{
                new double[]{0.1, 0.3, 0.6},
                new double[]{0.2, 0.5, 0.3},
```

```java
                new double[]{0.6, 0.2, 0.2}
        });
        System.out.println(pMatrix);

        System.out.println("P^2:");
        DoubleMatrix pSquared = pMatrix.multiply(pMatrix);
        System.out.println(pSquared);

        System.out.println("P^3:");
        DoubleMatrix pCubed = pSquared.multiply(pMatrix);
        System.out.println(pCubed);

        System.out.println("P^4:");
        DoubleMatrix pFourth = pCubed.multiply(pMatrix);
        System.out.println(pFourth);

        System.out.println("p3:");
        double[] p3 = pCubed.multiplyByLeft(p0);
        System.out.println(Arrays.toString(p3));

        System.out.println("p4:");
        double[] p4 = pFourth.multiplyByLeft(p0);
        System.out.println(Arrays.toString(p4));
    }
}
```

<u>Розв'язування</u>

Знайшли степені матриці P^2, P^3, P^4:

```
P^2:
[[0.43, 0.3, 0.27]
 [0.3, 0.37, 0.33]
 [0.22, 0.32000000000000006, 0.45999999999999996]]
P^3:
[[0.265, 0.333, 0.40199999999999997]
 [0.302, 0.341, 0.357]
 [0.362, 0.31800000000000006, 0.32000000000000006]]

P^4:
[[0.3343, 0.3264, 0.33930000000000005]
 [0.3126, 0.3325, 0.3549]
 [0.29180000000000006, 0.3316, 0.3766]]
```

Знайшли вектори розподілу ймовірностей:

$p_3 = p_0P^3$

```
p3:
[0.3186, 0.33020000000000005, 0.35120000000000007]
```

$p_4 = p_0P^4$

```
p4:
[0.30862000000000006, 0.33092, 0.36046]
```

Приклад №2:

```
[0.4, 0.2, 0.4]
[[0.4, 0.1, 0.5]
 [0.1, 0.6, 0.3]
 [0.3, 0.5, 0.2]]
P^2:
[[0.32000000000000006, 0.35, 0.33]
 [0.19, 0.52, 0.29]
 [0.22999999999999998, 0.42999999999999994, 0.33999999999999997]]
P^3:
[[0.262, 0.40700000000000003, 0.331]
 [0.215, 0.476, 0.309]
 [0.237, 0.45099999999999996, 0.31199999999999994]]
P^4:
[[0.24480000000000002, 0.43589999999999995, 0.31930000000000003]
 [0.2263, 0.4616, 0.3121]
 [0.23349999999999999, 0.4502999999999999, 0.3162]]
p3:
[0.2426, 0.4384, 0.319]
p4:
[0.23658, 0.4468, 0.31662]
```

Результат співпадає з результатом у документі *Зразок.docx* з точністю до 10^{-5}.

Приклад №3 (змінимо початковий розподіл ймовірностей):

```
double[] p0 = new double[]{1f/3, 1f/3, 1f/3};
```

```
[0.3333333432674408, 0.3333333432674408, 0.3333333432674408]
[[0.4, 0.1, 0.5]
 [0.1, 0.6, 0.3]
 [0.3, 0.5, 0.2]]
P^2:
[[0.32000000000000006, 0.35, 0.33]
 [0.19, 0.52, 0.29]
 [0.22999999999999998, 0.42999999999999994, 0.33999999999999997]]
P^3:
[[0.262, 0.40700000000000003, 0.331]
 [0.215, 0.476, 0.309]
 [0.237, 0.45099999999999996, 0.31199999999999994]]
P^4:
[[0.24480000000000002, 0.43589999999999995, 0.31930000000000003]
 [0.2263, 0.4616, 0.3121]
 [0.23349999999999999, 0.4502999999999999, 0.3162]]
p3:
[0.23800000709295271, 0.444666679918766, 0.3173333427906036]
p4:
[0.2348666736662388, 0.4492666800558566, 0.3158666760802269]
```

Список використаних джерел

Конспект з лекцій курсу “Математичні основи захисту інформації” Кривий С.Л.