Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Фізико-технічний інститут

ЛАБОРАТОРНА РОБОТА №4

з дисципліни

«Криптографія»

на тему: «Побудова регістрів зсуву з лінійним зворотним зв'язком та дослідження їх властивостей»

Виконав:

студент 3 курсу ФТІ

групи ФБ-74

Сизов Ігор

Перевірили:

Чорний О.

Савчук М. М.

Завадська Л. О.

Мета роботи

Ознайомлення з принципами побудови регістрів зсуву з лінійним зворотним зв'язком; практичне освоєння їх програмної реалізації; дослідження властивостей лінійних рекурентних послідовностей та їх залежності від властивостей характеристичного полінома регістра.

Порядок виконання роботи

0. Уважно прочитати методичні вказівки до виконання комп'ютерного практикуму.

1. Вибрати свій варіант завдання згідно зі списком. Варіанти завдань містяться у файлі Crypto_CP4 LFSR_Var.

2. За даними характеристичними многочленами p1(x), p2(x) скласти лінійні рекурентні співвідношення для ЛРЗ, що задаються цими характеристичними многочленами.

3. Написати програми роботи кожного з ЛРЗ 1 L , 2 L .

4. За допомогою цих програм згенерувати імпульсні функції для кожного з ЛРЗ і підрахувати їх періоди.

5. За отриманими результатами зробити висновки щодо властивостей кожного з характеристичних многочленів p1(x), p2(x): многочлен примітивний над 2 F ; не примітивний, але може бути незвідним; звідний. 6. Для кожної з двох імпульсних функцій обчислити розподіл k-грам на періоді, k≤ni, де ni - степінь полінома fi(x), i=1,2 а також значення функції автокореляції A(d) для 0≤ d ≤ 10. За результатами зробити висновки.

Варіант 14

$$P1(X)= X^{24} + X^{22} + X^{21} + X^{19} + X^{17} + X^{15} + X^{11} + X^{10} + X^{8} + X^{2} + 1$$

$$P2(X)= X^{20} + X^{19} + X^{18} + X^{14} + X^{12} + X^{10} + X^{9} + X^{7} + 1$$

Період

P1 2396745

P2 1048575

P1 k-грамми

1-gramms

0 - 0,49980160592804 1 - 0,50019839407196

2-gramms

00 - 0,249714612824732 01 - 0,249789714712961

11 - 0,250110983901493 10 - 0,250384688560814

3-gramms

000 - 0,124612756050393 001 - 0,125329978783725

011 - 0,125249870136372 110 - 0,125249870136372

111 - 0,124769218252255 010 - 0,124769218252255

100 - 0,124849326899608 101 - 0,125169761489019

4-gramms

0000 - 0,0626883805696395 0001 - 0,0617571171556078

0111 - 0,0628702940322371 1000 - 0,0625782311335712

1011 - 0,0629470648513149 1111 - 0,0623896419475755

1001 - 0,0629220308885722 0010 - 0,0625214874846876

1010 - 0,0625548661016779 0011 - 0,0628352464843972

1100 - 0,0620558557776717 0101 - 0,0622928439583034

1101 - 0,0625915825803674 1110 - 0,0621426401818467

0100 - 0,0623045264742501 0110 - 0,0625481903782799

5-gramms

00000 - 0,0314030069949035 00010 - 0,0311255473569362

11110 - 0,0310921687538724 00101 - 0,0307917613262988

11111 - 0,0313258189753186 10010 - 0,0313591975783823

00100 - 0,0311589259599999 10101 - 0,0314927119906373

10001 - 0,0311589259599999 01100 - 0,0310921687538724

01101 - 0,0313258189753186 00111 - 0,0311589259599999

01011 - 0,0316929836090197 11001 - 0,0312924403722549

00110 - 0,0313258189753186 00001 - 0,0311923045630637

10111 - 0,0312590617691911 01000 - 0,0314927119906373

11101 - 0,0313591975783823 11011 - 0,0314593333875736

01010 - 0,0311255473569362 01111 - 0,0313591975783823

00011 - 0,0312256831661274 10011 - 0,0312590617691911

01110 - 0,0310587901508087 10100 - 0,0313925761814461

11000 - 0,0311255473569362 11010 - 0,0314927119906373

01001 - 0,0310587901508087 10110 - 0,0310921687538724

10000 - 0,0311923045630637 11100 - 0,0310587901508087

P2 k-gramms

1-gramms

0 - 0,499999523162387 1 - 0,500000476837613

2-gramms

00 - 0,249895572463174 01 - 0,249653338724782

10 - 0,25055551634887 11 - 0,249895572463174

3-gramms

000 - 0,125121235963093 011 - 0,124145626206995

001 - 0,124878048780488 100 - 0,124878048780488

101 - 0,12561047135398 111 - 0,124878048780488

010 - 0,124878048780488 110 - 0,12561047135398

4-gramms

0000 - 0,0620653612722827 0001 - 0,0631411100048447

1011 - 0,0630571863448576 0010 - 0,0626032356385637

1110 - 0,0631296658693919 0101 - 0,0622255791686217

1010 - 0,0623514646586024 0011 - 0,0621149525259114

1111 - 0,0624086853358663 0100 - 0,0626108650621989

1000 - 0,0627672682467203 0110 - 0,06195854934139

1001 - 0,062893153736701 1100 - 0,0624201294713191

1101 - 0,0622255791686217 0111 - 0,0620272141541067

5-gramms

00000 - 0,0312185585198961 00001 - 0,0309753713372911

10110 - 0,0306701952650025 01011 - 0,0315857234818683

10010 - 0,0309753713372911 11010 - 0,0312805474095797

00111 - 0,0315857234818683 11101 - 0,0306701952650025

00100 - 0,0318908995541568 11100 - 0,0312805474095797

10100 - 0,0315857234818683 10000 - 0,0312805474095797

01110 - 0,0318908995541568 01101 - 0,0315857234818683

11000 - 0,0309753713372911 01100 - 0,0309753713372911

01010 - 0,0315857234818683 11110 - 0,0309753713372911

00110 - 0,0309753713372911 00101 - 0,0312805474095797

11001 - 0,0309753713372911 10111 - 0,0312805474095797

10011 - 0,0309753713372911 10101 - 0,0309753713372911

10001 - 0,0312805474095797 11111 - 0,0315857234818683

11011 - 0,0312805474095797 00010 - 0,0306701952650025

01000 - 0,0312805474095797 01001 - 0,0312805474095797

00011 - 0,0318908995541568 01111 - 0,0312805474095797

FIRST

d=1 1198847

d=2 1198463

d=3 1196671

d=4 1198463

d=5 1198463

d=6 1198463

d=7 1198847

d=8 1198847

d=9 1196671

d=10 1198847

SECOND

d=1 524287

d=2 524287

d=3 524287

d=4 524287

d=5 524287

d=6 524287

d=7 524287

d=8 524287

d=9 524287

d=10 524287

P1-незвідний характеристичний поліном

P2 – примітивний характеристичний поліном

Висновок

Ознайомився з принципами побудови регістрів зсуву з лінійним зворотним зв’язком; практично освоїв їх програмну реалізацію; провів дослідження властивостей лінійних рекурентних послідовностей та їх залежності від властивостей характеристичного полінома регістра.

Код

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CriptoLab4
{
    class Program
    {
        static int[] P1 = new int[] { 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1,
0 }; //24
        static int[] P2 = new int[] { 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1 };//20
        static int[] zeroOne = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1 }; //24
        static int[] zeroOneCpy = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1 }; //24
```

```
        static int[] zeroTwo = new int[] {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 };
//20
        static int[] zeroTwoCpy = new int[] {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 };
//20
        static int period = 0;
        static int period2 = 0;
        //
        static int[] buff1 = new int[5000000];
        static int buffcounter = 0;
        static int[] buff2 = new int[5000000];
        static int buff2counter = 0;

        public static int mod(int value, int modul)
        {
            while (value < 0 || value >= modul)
            {
                if (value < 0) value = value + modul;
                if (value >= modul) value = value - modul;
            }
            return value;
        }
        public static bool equals(int[]a, int[]b)
        {
            for(int i = 0; i < a.Length; i++)
            {
                if (a[i] != b[i]) return true;
            }
            return false;
        }
        public static void raschet(int[] a)
        {
            int NewNumb = 0;
            for(int i = 0; i < a.Length; i++)
            {
                NewNumb += a[i] * P1[i];
            }
            NewNumb = mod(NewNumb, 2);


            //buff += a[0].ToString();
            buff1[buffcounter] = a[0];
            buffcounter++;

            for(int i = 1; i < a.Length; i++)
            {
                a[i - 1] = a[i];
            }
            a[a.Length - 1] = NewNumb;


        }
        public static void raschet2(int[] a)
        {
            int NewNumb = 0;
            for (int i = 0; i < a.Length; i++)
            {
                NewNumb += a[i] * P2[i];
            }
            NewNumb = mod(NewNumb, 2);


            //buff += a[0].ToString();
            buff2[buff2counter] = a[0];
            buff2counter++;

            for (int i = 1; i < a.Length; i++)
            {
                a[i - 1] = a[i];
            }
            a[a.Length - 1] = NewNumb;


        }

        public static void kgramms(int k,int[] text,int length)
        {
            string[] kgramms = new string[500];
```

```
    int[] kgrammsKolvo = new int[500];
    int counter = 0;
    int cycle = 0;
    int dlina = length - (length % k);

    for(int i = 0; i < dlina; i += k)
    {
        string zz = "";
        bool trigg = false;
        for(int j = 0; j < k; j++)
        {
            zz += text[i + j].ToString();
        }

        for(int z = 0; z < counter; z++)
        {
            if (kgramms[z] == zz)
            {
                kgrammsKolvo[z]++;
                trigg = true;
                break;
            }
        }
        if (trigg == false)
        {
            kgramms[counter] = zz;
            kgrammsKolvo[counter] = 1;
            counter++;
        }
        cycle++;
    }
    Console.WriteLine($"{k}-gramms");
    int perehod = 0;
    for(int j = 0; j < counter; j++)
    {
        double chastot = kgrammsKolvo[j] / (double)cycle;

        Console.Write($"{kgramms[j]} - {chastot}               ");
        perehod++;
        if (perehod == 2)
        {
            Console.Write("\n");
            perehod = 0;
        }
    }
}

public static void autocorelation(int d,int[] text, int length)
{
    int summa = 0;
    for(int i = 0; i < length - 1; i++)
    {
        summa += mod((text[i] + text[mod((i + d),length)]), 2);
    }
    Console.WriteLine($"d={d}  {summa}");
}




static void Main(string[] args)
{



    do
    {
        raschet(zeroOne);
        period++;
        //if (period % 1000 == 0) Console.WriteLine(period);
    } while (equals(zeroOne, zeroOneCpy));
    do
    {
        raschet2(zeroTwo);
        period2++;
        //if (period2 % 1000 == 0) Console.WriteLine(period);
    } while (equals(zeroTwo, zeroTwoCpy));
```

```
            Console.WriteLine($"FIRST PERIOD {period}");
            Console.WriteLine($"SECOND PERIOD {period2}");
            Console.WriteLine("");
            Console.WriteLine("THE FIRST");
            for (int i = 1; i <6; i++)
            {
                kgramms(i, buff1, buffcounter);
            }
            Console.WriteLine("THE SECOND");
            for (int i = 1; i < 6; i++)
            {
                kgramms(i, buff2, buff2counter);
            }
            Console.WriteLine("AUTOCORRELATION");
            Console.WriteLine("\n FIRST");
            for (int i = 1; i < 11; i++)
            {
                autocorelation(i, buff1, buffcounter);
            }
            Console.WriteLine("\n SECOND");
            for (int i = 1; i < 11; i++)
            {
                autocorelation(i, buff2, buff2counter);
            }
            Console.ReadKey();
        }
    }

}
```