Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Фізико-технічний інститут

КРИПТОГРАФІЯ
КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

Побудова регістрів зсуву з лінійним зворотним зв’язком та дослідження їх властивостей

Виконали:

Студенти групи ФБ-73
Мухамедзянов Артем
Лень Олександр

Київ-2019

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 L1, L2.

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

5. За отриманими результатами зробити висновки щодо влавстивостей кожного з характеристичних многочленів p1(x), p2(x): многочлен примітивний над F2; не примітивний, але може бути незвідним; звідний.

6. Для кожної з двох імпульсних функцій обчислити розподіл k-грам на періоді, $k \leq ni$, де ni - степінь полінома fi(x), i=1,2 а також значення функції автокореляції A(d) для $0 \leq d \leq 10$. За результатами зробити висновки.

Код програми

```
#include <fstream>
#include <iostream>
#include <cstring>
#include <math.h>
#include <cstdio>
using namespace std;
struct    s_data
{
int *var25;
int  *def25;
int  *dif25;
int  *var;
int  *def;
int *dif;
};

char *get_t(s_data data, int &T, int size)
{
 cout << "Searching T\n";
 int suma = 0;
 char *mas = new char[100000000];
 int i = -1;
 while (++i < size)
 suma+=(data.var[i]*data.def[i]);
 //sdvig
 mas[T]=data.def[0];
 i = -1;
 while (++i < size - 1)
 data.def[i] = data.def[i+1];
 data.def[size - 1]=suma%2;
 T++;
 while(equal(data.def,data.def+size,data.dif) == 0)
 {
 suma = 0;
```

```
 i = -1;
 while (++i < size)
 suma+=(data.var[i]*data.def[i]);
 mas[T]=data.def[0];
 for(int i=0;i<size - 1;i++)
 {
 data.def[i]=data.def[i+1];
 }
 data.def[size - 1]=suma%2;
 T++;
   }
   cout<<"T = "<<T<<endl;
   return (mas);
}

void recur(int T, char *mas, int grams, int *num, int cur_num)
{
 if (cur_num == grams)
 {
 int n[5];
 memcpy(n, num, 5 * sizeof(int));
 for (int l = 0; l < grams; l++)
 cout << n[l] << " ";
 long int counter = 0;
 int i = 0 - grams;
 while ((i = i + grams) < T-T % grams)
 {
 char m[5];
 memcpy(m, mas + i, grams);
 if(equal(n, n + grams, m))
 counter++;
 }
 cout << counter << endl;
 return ;
 }
```

```
 num[cur_num] = 0;
 recur(T, mas, grams, num, cur_num + 1);
 num[cur_num] = 1;
 recur(T, mas, grams, num, cur_num + 1);
}

void calc_autocorrelation(int T, char *mas)
{
 cout << "calculating correlation\n"; //
 int k;
 int d = -1;
 int i;
 while (++d < 11 && !(k = 0) && (i = -1))
 {
 while (++i < T)
 k += (mas[i] + mas[(i + d) % T]) % 2;
 printf("%d %d\n", d, k);
 }
}

int main()
{
 s_data kripta;
 int var25[25]={1,0,0,1,0,0,1,1,1,0,1,0,0,1,1,1,0,1,0,1,1,0,0,0,0};
 int def25[25]={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};
 int dif25[25]={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};
 int var[20]={1,0,1,0,1,0,1,0,0,1,1,1,0,1,0,1,0,1,0,0};
 int def[20]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1};
 int dif[20]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1};
 kripta.var25 = var25;
 kripta.def = def;
 kripta.def25 = def25;
 kripta.var = var;
 kripta.dif = dif;
 kripta.dif25 = dif25;
```

```
 cout << "counting polinom 20-size\n";
 int T = 0;
 char *mas = get_t(kripta, T, 20);
 int mas_int[5];
 cout << "2-gram\n";
 recur(T, mas, 2, mas_int, 0);
 cout << "3-gram\n";
 recur(T, mas, 3, mas_int, 0);
 cout << "4-gram\n";
 recur(T, mas, 4, mas_int, 0);
 cout << "5-gram\n";
 recur(T, mas, 5, mas_int, 0);
 calc_autocorrelation(T, mas);
 delete[] mas;
 cout << "counting polinom 25-size\n";
 kripta.def = kripta.def25;
 kripta.dif = kripta.dif25;
 kripta.var = kripta.var25;
      T = 0;
 mas = get_t(kripta, T, 25);
 mas_int[5];
 cout << "2-gram\n";
 recur(T, mas, 2, mas_int, 0);
 cout << "3-gram\n";
 recur(T, mas, 3, mas_int, 0);
 cout << "4-gram\n";
 recur(T, mas, 4, mas_int, 0);
 cout << "5-gram\n";
 recur(T, mas, 5, mas_int, 0);
 calc_autocorrelation(T, mas);
 delete[] mas;
}
```

Результати роботи програми

Counting polinom 20-size

Searching T

T = 11275

2-gram

0 0 1394

0 1 1369

1 0 1438

1 1 1436

3-gram

0 0 0 459

0 0 1 468

0 1 0 453

0 1 1 534

1 0 0 462

1 0 1 446

1 1 0 472

1 1 1 464

4-gram

0 0 0 0 171

0 0 0 1 170

0 0 1 0 200

0 0 1 1 168

0 1 0 0 151

0 1 0 1 172

0 1 1 0 183

0 1 1 1 200

1 0 0 0 184

1 0 0 1 156

1 0 1 0 176

1 0 1 1 190

1 1 0 0 179

1 1 0 1 165

1 1 1 0 172

1 1 1 1 181

5-gram

0 0 0 0 0 74

0 0 0 0 1 70

0 0 0 1 0 67

0 0 0 1 1 64

0 0 1 0 0 68

0 0 1 0 1 63

0 0 1 1 0 70

0 0 1 1 1 71

0 1 0 0 0 79

0 1 0 0 1 54

0 1 0 1 0 75

0 1 0 1 1 60

0 1 1 0 0 74

0 1 1 0 1 61

0 1 1 1 0 80

0 1 1 1 1 73

1 0 0 0 0 70

1 0 0 0 1 57

1 0 0 1 0 80

1 0 0 1 1 65

1 0 1 0 0 69

1 0 1 0 1 72

1 0 1 1 0 77

1 0 1 1 1 66

1 1 0 0 0 68

1 1 0 0 1 75

1 1 0 1 0 70

1 1 0 1 1 75

1 1 1 0 0 85

1 1 1 0 1 64

1 1 1 1 0 77

1 1 1 1 1 82

calculating correlation

0 0

1 5608

2 5648

3 5616

4 5608

5 5648

6 5624

7 5680

8 5760

9 5624

10 5624

counting polinom 25-size

Searching T

T = 33554431

2-gram

0 0 4193298

0 1 4193480

1 0 4197138

1 1 4193299

3-gram

0 0 0 1398938

0 0 1 1398208

0 1 0 1395959

0 1 1 1399114

1 0 0 1398097

1 0 1 1398915

1 1 0 1397843

1 1 1 1397736

4-gram

0 0 0 0 523891

0 0 0 1 522991

0 0 1 0 523966

0 0 1 1 524799

0 1 0 0 524641

0 1 0 1 524003

0 1 1 0 524613

0 1 1 1 523950

1 0 0 0 525269

1 0 0 1 524382

1 0 1 0 524530

1 0 1 1 524387

1 1 0 0 523850

1 1 0 1 524896

1 1 1 0 525461

1 1 1 1 522978

5-gram

0 0 0 0 0 209659

0 0 0 0 1 209787

0 0 0 1 0 210040

0 0 0 1 1 209553

0 0 1 0 0 209425

0 0 1 0 1 209512

0 0 1 1 0 209614

0 0 1 1 1 210409

0 1 0 0 0 209531

0 1 0 0 1 209197

0 1 0 1 0 209576

0 1 0 1 1 209396

0 1 1 0 0 210106

0 1 1 0 1 210623

0 1 1 1 0 209261

0 1 1 1 1 209964

1 0 0 0 0 209387

1 0 0 0 1 209797

1 0 0 1 0 210032

1 0 0 1 1 209149

1 0 1 0 0 210216

1 0 1 0 1 209236

1 0 1 1 0 209960

1 0 1 1 1 209308

1 1 0 0 0 210149

1 1 0 0 1 209527

1 1 0 1 0 209600

1 1 0 1 1 209216

1 1 1 0 0 210357

1 1 1 0 1 209696

1 1 1 1 0 209763

1 1 1 1 1 209840

calculating correlation

0 0

1 16777216

2 16777216

3 16777216

4 16777216

5 16777216

6 16777216

7 16777216

8 16777216

9 16777216

10 16777216