中国科学技术大学

本科学位论文

BM 算法的实现

作者姓名: 吴立凡

学科专业: 网络空间安全

导师姓名: 李卫海

完成时间: 2020 年 5 月

University of Science and Technology of China

A dissertation for bachelor's degree

Implementation of BM algorithm

Author :	Lifan Wu
Speciality :	Cyberspace Security
Supervisor :	Weihai Li
Finished Time :	May, 2020

致谢

感谢 CSDN 的前辈们，他们的论文博客为本次实验提供了丰富的经验与建议。如果不是他们的帮助，这篇文章的完成也会变得无比艰难。

另外也要感谢所有教过我的老师们，他们所教授的许多内容都成为了我实践的坚实基础。在此特别感谢李卫海老师，他所教授的密码学导论为这篇论文提供了莫大的帮助。

目录

摘要

在实时加密（如实时通信）的情况下，往往需要一个流密码产生器提供加密序列。多数流密码的基本构造模块为反馈移位寄存器 LFSR。由于 LSFR 非常适合于硬件实现、可以产生大周期序列等优点，在实践中得到了广泛的应用。我们也可以用 LFSR 的长度来描述一个比特串的线性复杂度，一段有限二元序列 sn 的线性复杂度 L(sn)定义为：生成以 sn 为开始的二元序列的最短 LFSR 的长度。本次研究实现了 BM 算法，一个用于求解最短 LFSR 的长度的算法。并计算了在题目 2 中加密后的一段密文的线性复杂度

关键词：

BM 算法 LFSR 线性复杂度 随机字符串 联结多项式

Abstract

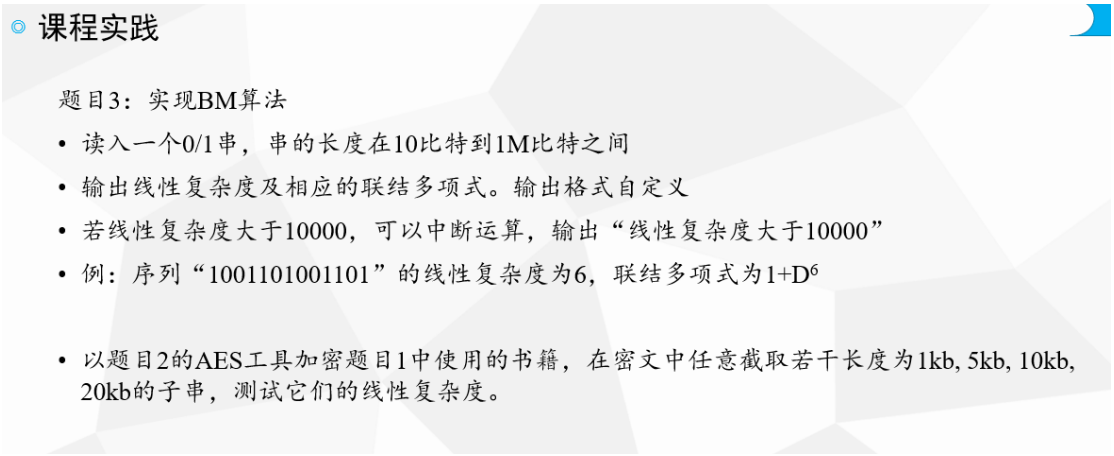
In the case of real-time encryption (such as real-time communication), a stream cipher generator is often required to provide an encryption sequence. The basic building block of most stream ciphers is the feedback shift register LFSR. Because LSFR is very suitable for hardware implementation and can generate large-period sequences, it has been widely used in practice. We can also use the length of LFSR to describe the linear complexity of a bit string. The linear complexity of a finite binary sequence sn is defined as the length of the shortest LFSR that generates a binary sequence starting with sn. This research implements the BM algorithm, an algorithm for solving the length of the shortest LFSR. And calculated the linear complexity of a piece of ciphertext encrypted in question 2.

Keywords:

BM algorithm LFSR linear complexity random string connection polynomial

正文

实验要求



实现效果

我们先看一下代码运行起来的实现效果：

```
wusar@LAPTOP-31I88NQC:~/crypt_homework$ make
g++ BM.cpp -o BM
./BM
0 0 1 0 1 1 0 1
L:4
1 1 1 0 0
```

可以看出，序列 00101101 的线性复杂度是 4，联结多项式是 1+D+D^2.

算法简介

（本段内容摘自 Crypt06-序列密码.pptx 李卫海老师）

若某 LFSR 在某初始状态下输出序列的前 n 项为 sn 的话，则称该 LFSR 生成有限序列 sn。

可以设想用 LFSR 来描述序列的复杂程度。LFSR 是线性的，因此它描述的是线性的复杂程度。

BM 算法用于求解最小阶数的线性反馈移位寄存器，使之输出的前 n 个比特与目标序列相同。

其时间复杂度为 O(2n)次比特操作。

线性复杂度的性质：令 s 和 t 都为二元序列

对任意 n≥1，子序列 sn 的线性复杂度满足 0≤L(sn)≤n

若 s 的周期为 N，则 L(s)≤N

L(sn)=0，当且仅当 sn 是长度为 n 的零序列

L(sn)=n，当且仅当 sn=0,0,0,…,0,1

算法流程

（图片来源：Crypt06-序列密码.pptx 李卫海老师）

具体实现

本实验所用到的所有代码都在附件中，主程序为 main.c，采用面向对象的编程思想，将 LFSR 封装成了一个类：

```
class LFSR
{
public:
```

```
    int L;
    int C[MAX_LENGTH], S[MAX_LENGTH];
public:
    LFSR(/* args */);
    ~LFSR();
    void init_LFSR(int L, int C[], int S[]);
    void print();
    int LFSR_next();
    int distance(int input);
    void BM(int sequence[],int length);
};
```

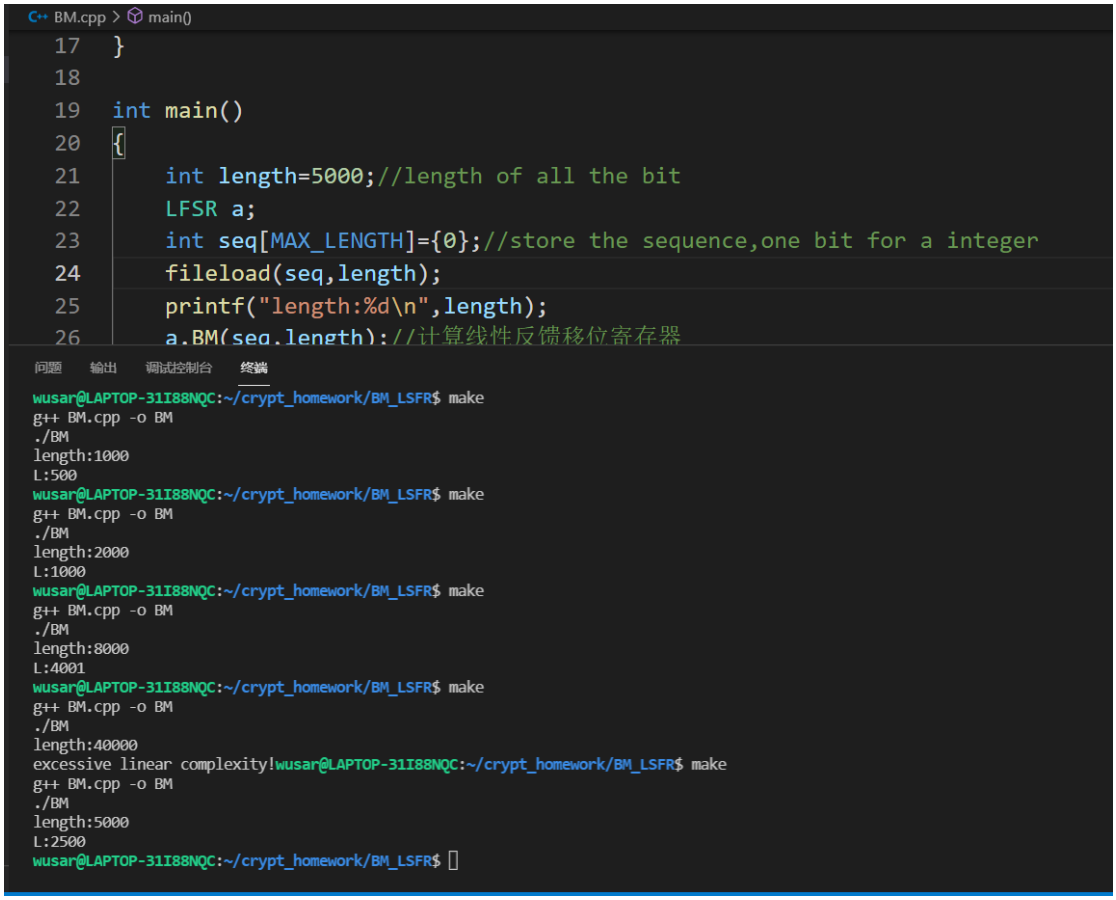
类的成员 L 表示移位寄存器的长度,C 表示连接多项式,S 表示移位寄存器的值，注意这里的 S 是一个逆序数组,S[0]=sL-1,S[L-1]=s0=输出,C[0]=1,C[1]=C1,C[]长度为 L+1。

init_LFSR 方法初始化移位寄存器，print 方法打印 LFSR 的连接多项式，LFSR_next()方法将移位寄存器进位并输出。distance 方法根据计算 BM 算法第二步中的 d。

成员函数 void BM(int sequence[],int length)传入一个 sequence 序列与其长度 length，从而使用 BM 算法算出移位寄存器 LFSR 的 L、C、S。

密文的线性复杂度：

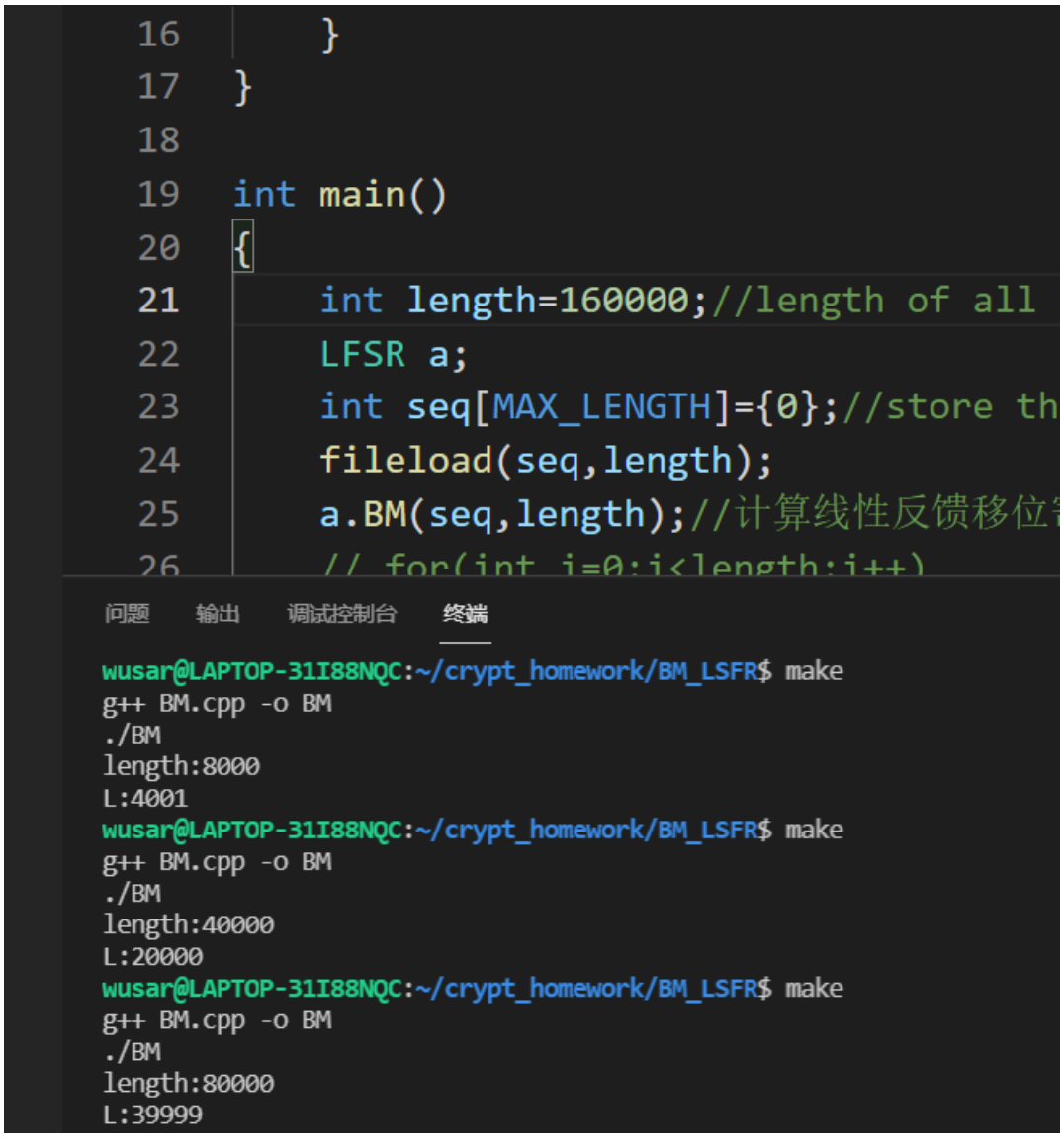
编写代码读取密文中一段比特串，并计算其线性复杂度，如图所示：



length 是比特串的长度（单位：bit），L 是线性移位寄存器的长度，我们发现线性移位寄存器的长度几乎就是比特串长度的一半。查阅资料可知，对于随机序列来说，其线性复杂度的均值就是比特串长度的一半。由于 AES 加密后的密文随机性很好，所以其理论上的线性复杂度就是密文长度的一半。在密文长度大于 20000bit 时，程序也输出了 excessive linear complexity!并终止了运算。

明文的线性复杂度

接下来，我们测试一下明文的线性复杂度：



我们可以看到，明文的线性复杂度也大约是明文长度的一半！但事实上明文并不是一种随机序列。这说明线性复杂度也许不是一种合适的描述文字序列的随机性的度量标准。（注：在进行此次实验时没有在线性复杂度过大时进行中断）