信息安全数学基础

实验指导书

教师：韩琦教授

实验地点：格物楼 213

实验时间：2023.11.18

一、背景介绍

有限域亦称伽罗瓦域（Galois Fields），是伽罗瓦于 18 世纪 30 年代研究代数方程根式求解问题时引出的概念。有限域在密码学、近代编码、计算机理论、组合数学等方面有着广泛的应用。

本次实验要求同学们利用 Java 或者其他编程语言对有限域四则运算算法和有限域欧几里得算法进行实践。

本次实验的目的包括：

1. 理解和掌握有限域的概念，以及有限域在计算机领域的应用。

2. 了解有限域四则运算算法和有限域欧几里得算法的流程, 将其和整数四则运算以及整数欧几里得算法进行对比。

3. 编码实践有限域四则运算算法和有限域欧几里得算法。

二、实验环境

- 操作系统：Windows7、Windows10、Windows11
- 编程语言：Java
- 代码编辑器：Eclipse

Ps：同学可以选择自己常用的代码编辑器。

三、实验内容

3.1 有限域四则运算算法

在抽象代数中，域是一个对加法和乘法封闭的集合，其中要求每个元素都有加法逆元，每个非零元素都有乘法逆元。若域 F 只包含有限个元素，则称为有限域，有限域中元素的个数称为有限域的阶。可以证明，每个有限域的阶必为素数的幂，即有限域的阶可表示为 p^n（p 是素数，n 是正整数）。有限域通常记为 $GF(p^n)$。

有限域 $GF(p^n)$中的元素可以看成有限域 GF(p)上次数小于 n 的多项式，因此 $GF(p^n)$构成 GF(p)上的 n 维线性空间，其中的一组基为$\{1, x, x^2, x^3, ..., x^{n-1}\}$，所以有限域 $GF(p^n)$中的所有元素可以用基$\{1, x, x^2, x^3, ..., x^{n-1}\}$的线性组合来表示，其线性组合的系数在 GF(p)中，即 $GF(p^n)=\{a_0+a_1x+a_2x^2+a_3x^3+...+a_{n-1}x^{n-1} \mid a_i$属于 GF(p), $i = 0, 1, 2, ..., n-1\}$。

若将 GF(p)上的加法单位元记为 0，乘法单位元记为 1，元素 a 的加法逆元记作-a，非零元素 b 的乘法逆元记作 b^{-1}，则有：$a + (-a) = 0 \pmod p$，$b * b^{-1} = 1 \pmod p$。针对 GF(p)中的元素 a 和非零元素 b，加法是$(a + b) \bmod p$，减法是$(a + (-b)) \bmod p$，乘法是$(a * b) \bmod p$，除法是$(a * b^{-1}) \bmod p$，该算法对多项式的运算同样成立，因此 $GF(p^n)$上的四则运算可以由 GF(p)上多项式的四则运算导出。

特别地，当 p = 2 时，$GF(p^n)$中的元素 $a_0+a_1x+a_2x^2+a_3x^3+...+a_{n-1}x^{n-1}$ 可以简化为二进制数 $a_{n-1}...a_3a_2a_1a_0$。因为计算机中使用的是二进制数，且 1 字节为 8 位，所以在密码学中常常用到有限域 $GF(2^8)$。

3.1.1 加法运算

$GF(2^n)$上的加法即为 GF(2)上多项式的加法，具体是将 GF(2)上多项式的系数分别相加。而对于 GF(2)上的元素，加法即为异或运算。所以，$GF(p^n)$上的加法即为按位异或运算。

3.1.2 减法运算

$GF(2^n)$上的减法即为 GF(2)上多项式的减法，具体是将 GF(2)上多项式的系数分别相减。而对于 GF(2)上的元素，减法即为加法。所以，$GF(2^n)$上的减法即为 $GF(2^n)$上的加法。

3.1.3 乘法运算

GF(2^n)上的乘法即为 GF(2)上多项式的乘法，具体是将 GF(2)上的两个多项式相乘，但是两个多项式相乘之后的次数可能会大于等于 n，因此在计算两个多项式的乘积之后还需要模一个 GF(2)上的 n 次不可约多项式，来保证得到的多项式次数小于 n。

在具体实现中，模 n 次不可约多项式的操作可以分解到每一次乘 x 的运算中，即乘 x 的运算可通过左移一位后再根据条件按位异或给定的不可约多项式对应的 n 位二进制数的后 n-1 位实现。乘 x 的更高次幂可通过重复使用该方法来实现，这样一来，GF(2^n)上的乘法结果可由多个中间结果相加得到。

有限域 GF(2^8)上的乘法运算的详细步骤如下：

（1） 输入 a，b 和不可约多项式 poly，并将 a，b 转换成二进制数，将乘法结果 ans 初始化为 0；

（2） 判断 b 是否大于 0，若大于 0，则转第（3）步，否则转第（6）步；

（3） 判断 b 的最低位是否为 1，若是，则将 ans 与 a 进行按位异或运算，再将 a 左移一位，否则直接将 a 左移一位；

（4） 判断 a 的最高位是否为 1，若是，则将 a 与 poly 进行按位异或运算，再将 a 与 0xff 进行按位与运算，否则直接将 a 与 0xff 进行按位与运算；

（5） 将 b 右移一位，转第（2）步

（6） 输出 ans，算法结束。

3.1.4 除法运算

在本次实验中，GF(2^n)上的除法指 GF(2)上多项式的带余除法，具体是将 GF(2)上的两个多项式相除，得到商和余数。

在具体实现中，令 a 为被除数，b 为除数，q 为商，r 为余数，且均为二进制数形式。将 q 的值初始化为 0，当 a 的比特长度大于等于 b 的比特长度时，b 左移 k 位后与 a 长度相等，此时将 q 与 1 左移 k 位后的值相加，将 a 与 b 左移 k 位后的值相减，再判断 a 与 b 的大小；若 a 大于 b，则重复上述步骤，直至 a 小于 b，此时 a 的值即为 r。

有限域上除法运算的详细步骤如下：

（1） 输入 a，b，并将 a，b 转换为二进制数，将商 ans 初始化为 0；

（2） 判断 a 的比特长度是否大于等于 b 的比特长度，若是，则转第（3）步，否则转第（6）步；

（3） 计算 a 的比特长度与 b 的比特长度之差，并将运算结果赋值给 rec；

(4) 将 a 与 b 左移 rec 位后的值进行按位异或运算；

(5) 将 ans 与 1 左移 rec 位后的值进行按位异或运算，转第（2）步；

(6) 输出 ans，算法结束。

3.2 有限域欧几里得算法

有限域欧几里得算法与整数欧几里得算法类似，只需注意这里的四则运算和取模运算都是定义在 GF(2^8)上的。

求最大公约数的欧几里得算法的原理基 gcd(a, b) = gcd(b, a mod b)。假设 $a=bq_0 + r_0$，则有 gcd(a, b) = gcd(b, r_0)；同理可得，存在 q_1 和 r_1，使得 $b = r_0q_1 + r_1$，则有 gcd(b, r_0) = gcd(r_0, r_1)如此维续进行下去，直到 $r_n = 0$， gcd(a, b) = gcd(r_{n-1}, r_n) = r_{n-1}。

有限域下欧几里得算法的详细步骤如下：

(1) 输入 a，b；

(2) 判断 a mod b 是否为 0，若是，则转第（4）步，否则转第（3）步；

(3) 计算 r = a mod b，并令 a = b，b = r，转第（2）步；

(4) 输出 b，算法结束。

3.3 有限域求乘法逆元

在有限域 GF(p)上，如果对任意一个非零元素 g，有 ag + bp = gcd(g, p) = l，则 ag = 1(mod p)，因此 a 就是 g 的乘法逆元，求解 g 的逆元的过程就是求解 a 的过程，因此可以利用有限域扩展欧几里得算法，求出元素的乘法逆元。

请补全 FiniteField.java 代码中的方法，最终实现上述的算法，并且将代码的运行进行演示。

四、测试用例

完成实验，并通过下面的测试数据：

有限域四则运算	输入数据	运算结果
有限域加法	0x89 + 0x4d	0xc4
	0xaf + 0x3b	0x94
	0x35 + 0xc6	0xf3

有限域减法	0x89 - 0x4d	0xc4
	0x9f - 0x3b	0xa4
	0x35 - 0xc6	0xf3
有限域乘法	0xce * 0xf1	0xef
	0x70 * 0x99	0xa2
	0x00 * 0xa4	0x00
有限域除法	0xde / 0xc6	0x01
	0x8c / 0x0a	0x14
	0x3e / 0xa4	0x00

有限域四则运算测试数据

输入数据	最大公约数 gcd(a, b)
0x75, 0x35	0x01
0xac, 0x59	0x03
0xf8, 0x2e	0x02
0x48, 0x99	0x09

有限域欧几里得算法测试数据

输入数据	逆元
0x8c	0xf7
0xbe	0x86
0x01	0x01
0x2d	0x44

有限域求乘法逆元测试数据

完成实验后将实验结果存放在实验报告中，并且和实验源码一起打包发送给助教，提交时**注明姓名和学号**。