Міністерство освіти і науки України

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Навчально-науковий Фізико-технічний інститут

Спеціальні розділи обчислювальної математики

Комп'ютерний практикум №4

Реалізація операцій у скінченних полях характеристики 2
(нормальний базис)

Виконав:
студент групи ФІ-12
Юрчук Олексій

Київ 2024

Тема: Реалізація операцій у скінченних полях характеристики 2 в нормальному базисі

Мета: Одержання практичних навичок програмної реалізації обчислень у полі Галуа характеристики 2 в нормальному базисі; ознайомлення з прийомами ефективної реалізації критичних по часу ділянок програмного коду та методами оцінки їх ефективності.

Завдання:

А) Реалізувати поле Галуа характеристики 2 степеня m в нормальному базисі з операціями:

1) знаходження константи 0 – нейтрального елемента по операції «+»;
2) знаходження константи 1 – нейтрального елемента по операції «·»;
3) додавання елементів;
4) множення елементів;
5) обчислення сліду елементу;
6) піднесення елемента поля до квадрату;
7) піднесення елемента поля до довільного степеня (не вище $2^m - 1$, де m - степінь розширення);
8) знаходження оберненого елемента за множенням;
9) конвертування (переведення) елемента поля в $m -$ бітний рядок (строкове зображення) і навпаки, де m - степінь розширення;

Б) Проконтролювати коректність реалізації поля для кожної операції; наприклад, для декількох a, b, c, d перевірити тотожності

- $(a + b) \cdot c = b \cdot c + a \cdot c$
- $d^{2^{m-1}} = 1, d \neq 0$

В) Визначити середній час виконання операцій у полі. Підрахувати кількість тактів процесора (або інших одиниць виміру часу) на кожну операцію. Результати подати у вигляді таблиць або діаграм.

Теоретичні відомості:

Полем називається множина елементів з двома заданими на ній бінарними операціями, додаванням та множенням (+ та ·, інколи позначаються $\oplus$ та $\otimes$) для яких виконуються умови:

а) щодо операції додавання елементи поля утворюють абелеву групу з нейтральним елементом 0;

б) щодо операції множення всі елементи, окрім 0, також утворюють абелеву групу з нейтральним елементом 1;

в) додавання та множення пов'язані між собою законом дистрибутивності: для будь-яких елементів поля x, y, z виконується $x(y+z) = xy + xz$.

Число елементів поля називається *порядком* поля. Поле називається *скінченним* (або *полем Галуа*), якщо воно має скінченну кількість елементів. Скінченне поле порядку q позначається $GF(q)$ або F_q. Порядок скінченного поля завжди є степенем деякого простого числа, $q = p^m$, число m називається *степенем* поля, а просте число p – його *характеристикою*.

Многочленом $f(t)$ степеня m над полем $GF(p)$ є вираз вигляду

$$f(t) = a_m t^m + a_{m-1} t^{m-1} + \cdots + a_1 t + a_0,$$

де коефіцієнти многочлена $a_i \in GF(p)$, $i = 0, \ldots m$, а t – змінна, деякий символ, що не належить полю.

2.1. Нормальні базиси скінченних полів характеристики 2

Розглянемо скінчене поле $GF(p^m)$. Якщо x – такий елемент поля $GF(p^m)$, що елементи $\{x, x^p, x^{p^2}, \ldots, x^{p^{m-1}}\}$ лінійно незалежні над $GF(p)$, то ці елементи утворюють базис поля $GF(p^m)$, який називається *нормальним*. Доведено, що нормальний базис існує для довільного скінченного поля.

У полях $GF(2^m)$ для багатьох значень m існує *гаусівський оптимальний нормальний базис* (він є частковим випадком нормального базису). Ми будемо розглядати (згідно ДСТУ 4145-2002) поля, які мають гаусівський оптимальний нормальний базис другого типу, що має місце, якщо число $p = 2m+1$ просте і для найменшого натурального числа k, такого, що $2^k \equiv 1 (\mathrm{mod}\ p)$, виконується одна з наступних умов:

а) $k = 2m$;

б) $p \equiv 3 (\mathrm{mod}\ 4)$ і $k = m$.

Надалі гаусівський оптимальний нормальний базис типу 2 будемо називати просто *оптимальним нормальним базисом* ***(ОНБ)***.

- Операції у нормальному базисі для поля характеристики 2

А) Додавання

2.2.1. Додавання в ОНБ

Додавання в ОНБ виконується так само, як і в поліноміальному базисі – покомпонентно (побітово).

Б) Множення

2.2.4. Множення в ОНБ

Добуток $z = u \cdot v$ елементів $u = (u_0, u_1, \ldots, u_{m-1})$ та $v = (v_0, v_1, \ldots, v_{m-1})$ в ОНБ обчислюється за формулою

$$z_i = (u <<< i) \cdot \Lambda \cdot (v <<< i)^{\mathrm{T}} =$$
$$= (u_i, u_{i+1}, \ldots, u_{m-1}, u_0, u_1, \ldots, u_{i-1}) \cdot \Lambda \cdot (v_i, v_{i+1}, \ldots, v_{m-1}, v_0, v_1, \ldots, v_{i-1})^{\mathrm{T}},$$

де $<<< i$ позначає циклічний зсув вліво на i компонент,

$^{\mathrm{T}}$ – знак транспонування,

Λ – мультиплікативна матриця розмірності m на m,

$z = (z_0, z_1, \ldots, z_{m-1})$.

Складність множення визначається числом ненульових елементів у матриці Λ (як її обчислювати, написано в п. 2.5). В загальному випадку в цій матриці не менше $2m-1$ ненульових елементів. Якщо нормальний базис є оптимальним, то ненульових елементів рівно $2m-1$ (власне, з цієї причини такий базис і називається оптимальним). Повільні програмні реалізації (як ця лабораторна робота ☺) можуть оптимальність базису і не використовувати, а рахувати $u \Lambda v^T$ «в лоб».

2.2.5. Знаходження мультиплікативної матриці в ОНБ

Мультиплікативна матриця Λ складається з рядків, які є розкладом в ОНБ m добутків елементів базису вигляду $x \cdot x^{2^j}$, $j = 0, \dots, m-1$, тобто

$$\Lambda = \begin{bmatrix} x \cdot x \\ \dots \\ x \cdot x^{2^j} \\ \dots \\ x \cdot x^{2^{m-1}} \end{bmatrix}$$

Доведено, що матриця Λ не залежить від вибору ОНБ (бо він єдиний з точністю до циклічного зсуву).

Виявилося, що можна зовсім позбавитися від мови теорії скінченних полів і обчислювати мультиплікативну матрицю в ОНБ за такою простою формулою:

$$\lambda_{i,j} = 1\text{, якщо виконується одна з таких умов: } \left[\begin{array}{l} 2^i + 2^j \equiv 1 \bmod p \\ 2^i - 2^j \equiv 1 \bmod p \\ -2^i + 2^j \equiv 1 \bmod p \\ -2^i - 2^j \equiv 1 \bmod p \end{array}\right.,$$

$$\lambda_{i,j} = 0 \text{ в усіх інших випадках,}$$

де $p = 2m+1$, $0 \le i, j \le m-1$. Тепер все, що потрібно – це знати $2^i \bmod p$ для $0 \le i \le m-1$.

В) Піднесення до квадрату

2.2.2. Піднесення до квадрату в ОНБ

Перевага використання оптимального нормального базису особливо відчутна при виконанні операції піднесення до квадрата. Дійсно, для довільного елемента $y = \sum_{i=0}^{m-1} y_i x^{2^i} =$ $= (y_0, ..., y_{m-1})_{NB} \in GF(2^m)$ з того, що $y_i \in GF(2)$ та лінійності операції піднесення до квадрата у полі характеристики 2 випливає, що

$$y^2 = \left(\sum_{i=0}^{m-1} y_i x^{2^i}\right)^2 = \sum_{i=0}^{m-1} \left(y_i x^{2^i}\right)^2 = \sum_{i=0}^{m-1} y_i x^{2^{i+1}} = (y_{m-1}, y_0, ..., y_{m-2}),$$

$$\text{або } y^2 = (y >>> 1),$$

де >>> – циклічний зсув вправо

Отже, піднесення до квадрата в оптимальному нормальному базисі зводиться до циклічного зсуву вправо компонент векторного зображення елемента.

Г) Знаходження оберненого в нормальному базисі

2.2.6. Знаходження оберненого елемента в ОНБ

Обернений елемент в оптимальному нормальному базисі також можна знайти за формулою $y^{-1} = y^{2^m-2}$, $y \ne 0$, або за допомогою алгоритму Евкліда. Втім, для ОНБ був розроблений спеціальний алгоритм пошуку оберненого елементу, що використовує багато возведень до квадрату та порівняно малу кількість множень – це так званий алгоритм Іто-Цудзії.

Д) Знаходження сліду в нормальному базисі

2.2.3. Обчислення сліду елементу в ОНБ

Іншою операцією, яка ефективно виконується в ОБН, є обчислення сліду елементу. Дійсно, розглянемо елемент $y=(y_0,\dots,y_{m-1})_{NB}\in GF(2^m)$; тоді $tr(y)=y+y^2+y^4+\dots+y^{2^{m-1}}$. Однак з п. 2.2 маємо:

$$y=(y_0,y_1,y_2,\dots,y_{m-1}),$$
$$y^2=(y_{m-1},y_0,y_1,\dots,y_{m-2}),$$
$$y^4=(y_{m-2},y_{m-1},y_0,\dots,y_{m-3}),$$
$$\dots$$
$$y^{2^{m-1}}=(y_1,y_2,y_3,\dots,y_{m-1},y_0).$$

Звідси випливає, що $tr(y)=(c,c,\dots,c)_{NB}$, де $c=y_0+y_1+\dots+y_{m-1}$ (додавання виконується в полі $GF(2)$). Оскільки $(0,0,0,\dots,0)$ є зображенням нуля, а $(1,1,1,\dots,1)$ – зображенням одиниці в нормальному базисі, остаточно маємо:

$$tr(y)=y_0+y_1+\dots+y_{m-1}.$$

Таким чином, слід елементу дорівнює сумі коефіцієнтів його представлення у нормальному базисі.

Хід роботи

Введені числа:

a = `15795109c9b0e4f4f2f6ef8b1cbadc4adadaefcafff`

b = `0edf130b2a5bf30a816c4131bcdb4523caef5480134`

c = `16cdef39cbdb4523caef5480134edf130b2a5bf30a8`

d = `fffffffffffffffffffffffffffffffffffffffffff`

Перевірка властивостей:

```
Checking the properties:
First property:         (a+b)*c = a*c + b*c
13b21796ed4d4c74d75f189bc9f9b17317779c634000

13b21796ed4d4c74d75f189bc9f9b17317779c634000

Second property:        a^(2^(m-1)) = 1
1fffffffffffffffffffffffffffffffffffffffffff
```

, де `1fffffffffffffffffffffffffffffffffffffffffff` - нейтральний елемент за множенням, тобто 1.

Результат додавання, множення, піднесення до степеню за модулем незвідного полінома, знаходження сліду та оберненого елемента в полі:

```
SUM:    1ba64202e3eb17fe739aaebaa06199691035bb4aecb

Multiply:       1540560a0122a75d46f5ba1ecd3e4784454fa5f75ba2

Square: 10abca884e4d8727a797b77c58e5d6e256d6d77e57ff

Trace:  0

Inverse to a:   11054e3d83ff6d85e3445527f4496e4615f1f38c543

Degree: 1fdf04fa616d3709cd4d451f2d100b9fd7b68fc7e36b
```

Усі операції працюють коректно, перевірено за допомогою TestL2.exe

Також було виміряно середній час виконання кожної зазначеної операції, на 10 запусках, та відповідна кількість тактів процесора, результати були згруповані у таблицю:

Операція	Середній час виконання (секунди)	Кількість тактів процесора
Додавання	6.3e-07	819
Множення	0.160953	209238900
Піднесення в квадрат	9.7e-07	1261
Піднесення до степеня	11.2431	14616030000
Слід	3e-07	390
Обернений	1.35369	1759797000

Кількість тактів процесора обрахована за формулою:
кількість тактів = час виконання (секунди) * тактова частота (Гц)
Тактова частота комп’ютера: 1,30ГГц, тобто 1300000000Гц

Додатки:

Увесь код можна знайти за посиланням, на GitHub:
https://github.com/MansteinOrGuderian/SpRzOM-4