Efficient Implementation of Multiplication on Extension Field Using GPU

Tanaka, Satoshi
Kyushu University | Institute of Systems Information Technologies and Nanotechnologies

Yasuda, Takanori
Institute of Systems Information Technologies and Nanotechnologies

Sakurai, Koichi
Kyushu University | Institute of Systems Information Technologies and Nanotechnologies

出版情報：MI lecture note series. 53, pp.113-121, 2013-12-26. Institute of Mathematics for Industry, Kyushu University
バージョン：
権利関係：

Efficient Implementation of Multiplication on Extension Field Using GPU

Satoshi Tanaka[*,**], Takanori Yasuda[**], Kouichi Sakurai[*,**]

* : Kyushu University

** : Institute of Systems, Information Technologies and Nanotechnologies

Abstract:

Evaluating non-linear multivariate polynomial systems over finite fields is an important subroutine for encryption and signature verification in multivariate public-key cryptography (MPKC). The security of MPKC definitely becomes lower if a larger field is used instead of $GF(2)$ given the same number of bits in the key. However, we still would like to use larger fields because MPKC tends to run faster at the same level of security if a larger field is used. The heaviest computation of evaluating non-linear multivariate polynomial system is multiplication. Therefore, we must find the best way of multiplications.

Nowadays, graphics processing units (GPUs) have over 100 times computational power than CPU. They are constructed by hundreds cores. Hence, it seems that GPUs are suited as parallel general computing machines. Therefore, researchers applied parallel algorithms to GPUs.

In this work, we compare the efficiency of several techniques for multiplication methods over $GF(2^{16})$ via their implementations on a CPU and a GPU. In CPU implementations, Zech's method is fastest, and it multiplies 67,108,864 instances in 1.2 seconds. On the other hand, for GPU implementations, it seems that $GF(2^4)$ is a very efficient intermediary field for building extension fields over $GF(2^{16})$. The time of 67,108,864 multiplications is about 60.3 milliseconds. GPU implementations are about 20 times faster than CPU implementations.

Efficient Implementation of Multiplication on Extension Field Using GPU

Satoshi Tanaka, Takanori Yasuda, Kouichi Sakurai
†: Kyushu University
‡: Institute of Systems, Information Technologies and Nanotechnologies

Multivariate Public-Key Cryptography (MPKC)

- Using multivariate polynomial system for encryption

$$M = (x_1, \dots, x_n) \Rightarrow \begin{cases} y_1 = p_1(x_1, \dots, x_n) \\ \vdots \\ y_m = p_m(x_1, \dots, x_n) \end{cases} \Rightarrow C = (y_1, \dots, y_m)$$

Message

system

Ciphertext

Computational Cost of MPKC

- $\frac{m(n^2+3n)}{2}$ additions, $m(n^2+2n)$ multiplications, n variables, m polynomials,

Unknowns	Polynomials	Additions	Multiplications
40	60	51,600	100,800
60	90	170,100	334,800
80	120	398,400	787,200
128	256	2,146,304	4,259,840
256	512	16,973,824	33,816,576
320	640	33,075,200	65,945,600
512	1024	135,004,160	269,484,032

Extension Field of Finite Field

- Field extension K/F, $K = GF(p^k)$, $F = GF(p)$
  - Primitive polynomial $f_0(x)$:
    $f_0(x) = x^k + c_{k-1}x^{k-1} + \cdots + c_1x + c_0,\ (c_0, \ldots, c_{k-1} \in F)$
  - Element $e = (e_1, \ldots, e_k)$:
    e: $e(x) = e_{k-1}x^{k-1} + \cdots + e_1x + e_0, \quad (e_0, \ldots, e_{k-1} \in F)$
- Addition:
  $a + b = (a_0 + b_0 \bmod p, \ldots, a_{k-1} + b_{k-1} \bmod p)$

Multiplications on Extension Field

- Multiplication:

$$a * b = a(x) * b(x) \; mod \; f_0(x)$$
$$= \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} a_i b_j x^{i+j} \; mod \; f_0(x)$$

  - Computational Cost:
    $(k-1)^2$ additions, k^2 multiplications on F,
    1 modulo $f_0(x)$

Reduce Multiplication Cost

- Transform multiplication formula
- Zech's method
- Use multiplication table

Transform Multiplication Formula

- Precompute modulo $f_0(x)$
  - E.g. $GF(2^2)$, $f_0(x) = x^2 + x + 1 = 0$
    $c = a * b = (a_0, a_1) * (b_0, b_1)$:
    $c_0 = a_0 b_0 + a_1 b_1$
    $c_1 = a_1 b_0 + a_0 b_1 + a_1 b_1$
- Computational cost:
  $2(k-1)^2$ additions, k^2 multiplications,
  $(k-1)^2$ constant multiplications on F

Zech's method

- Precompute
  $x^t \bmod f_0(x), \qquad (0 \le t < p^k - 1)$
- $c = a * b = x^{t_a} * x^{t_b}$:
  $x^{(t_a + t_b) \bmod p^k - 1} \bmod f_0(x)$
  - E.g. . $GF(2^2)$, $f_0(x) = x^2 + x + 1 = 0$

elements	(0,0)	(0,1)	(1,0)	(1,1)
index	$-\infty$	0	1	2

- Computational cost:
  3 lookups, 1 additions, 1 modulo $(p^k - 1)$
- Memory cost: $2p^k \lceil k \log_2 p \rceil$ bits

Use Multiplication Table

- Precompute all multiplications on K
  - E.g. $GF(2^2), f_0(x) = x^2 + x + 1 = 0$

$*$	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
(0,1)	(0,0)	(0,1)	(1,0)	(1,1)
(1,0)	(0,0)	(1,0)	(1,1)	(0,1)
(1,1)	(0,0)	(1,1)	(0,1)	(1,0)

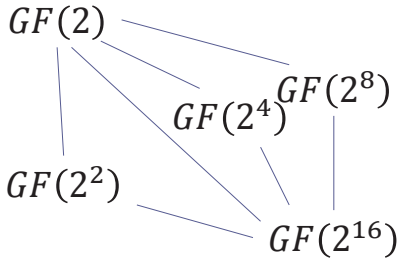
- Computational cost: 1 lookup
- Memory cost: $\left(p^k\right)^2 \lceil k \log_2 p \rceil$ bits

Which is the Best Way?

- Use multiplication table:
  - $GF(2^8)$: 64 kB
  - $GF(2^{16})$: 8 GB
- Zech's method:
  - $GF(2^{16})$: 256 kB
  - $GF(2^{32})$: 32 GB

Use Intermediary Field

- Intermediary field L of K/F
  - $L = GF(p^l)$, (l is a divisor of k)
- Multiplications on K with K/L, L/F:
  - $2(k/l-1)^2$ additions on L, $(k/l)^2$ multiplications on L
  - Multiplications on L:
    - Multiplication table
    - Zech's method



Example: Multiplication on $GF(2^{16})$

- $GF(2^{16})/GF(2^8)/GF(2)$:
  - $GF(2^8)/GF(2)$: Multiplication table
    $f_0(x) = x^8 + x^4 + x^3 + x^2 + 1 = 0$
  - $GF(2^{16})/GF(2^8)$: Polynomial
    $f_0(y) = y^2 + y + (x^5 + x) = 0,$

Multiplication on GPU

- Should parallelize about multiplication on GPU?
  - Should parallelize about evaluating multivariate polynomial system
    - Evaluating system needs many computaions
    - Reduce communications between CPU and GPU

Experimental Result of $GF(2^{16})$

- Computational time of 67,108,864 multiplications

CPU	Zech's method	Polynomial
$GF(2^{16})/GF(2)$	1,174.1971 ms	21,982.8976 ms
$GF(2^{16})/GF(2^2)/GF(2)$	N.A.	9,591.4002 ms
$GF(2^{16})/GF(2^4)/GF(2)$	N.A.	3,500.0024 ms
$GF(2^{16})/GF(2^8)/GF(2)$	N.A.	1,357.0016 ms

GPU	Zech's method	Polynomial
$GF(2^{16})/GF(2)$	88.5837 ms	346.5904 ms
$GF(2^{16})/GF(2^2)/GF(2)$	N.A.	149.7266 ms
$GF(2^{16})/GF(2^4)/GF(2)$	N.A.	60.3309 ms
$GF(2^{16})/GF(2^8)/GF(2)$	N.A.	92.5700 ms

Conclusion

- We present how to multiply on extension field
  - Show example of multiplications on $GF(2^{16})$, CPU, GPU implementation result
- Future works
  - Generalize to all finite fields of multiplications.
  - Apply to MPKC