A remark on the computation of cube roots in finite fields [⋆]

Nozomu Nishihara[⋆⋆1], Ryuichi Harasawa[1], Yutaka Sueyoshi[1], and Aichi Kudo[1]

Faculty of Engineering, Nagasaki University
1-14 Bunkyomachi, Nagasaki-shi, Nagasaki, 852-8521, Japan
`nozomu.nishihara@nms.necel.com`
`{harasawa, sueyoshi, kudo}@cis.nagasaki-u.ac.jp`

Abstract

We consider the computation of cube roots in finite fields. For the computation of square roots in finite fields, there are two typical methods; the Tonelli-Shanks method [9, 12] and the Cipolla-Lehmer method [4, 6]. The former can be extended easily to the case of r-th roots, which is called the Adleman-Manders-Miller method [1], but it seems to be difficult to extend the latter to more general cases. In this paper, we propose two explicit algorithms for realizing the Cipolla-Lehmer method in the case of cube roots for prime fields $\mathbb{F}_p$ with $p \equiv 1 \pmod 3$. We implement these methods and compare the results.

Keywords: cube root, finite field, the Tonelli-Shanks method, the Cipolla-Lehmer method

1 Introduction

Solving algebraic equations over finite fields is one of the most popular topics in the computational algebra. The typical equations are in the form $x^r = a$, given a natural integer $r(\geq 2)$ and an element a in the base field. For square roots (i.e., $r = 2$), there exist two well-known methods; the Tonelli-Shanks method [9, 12] and the Cipolla-Lehmer method [4, 6]. The former can be extended easily to the case of r-th roots, which is called the Adleman-Manders-Miller method [1], but it seems to be difficult to extend the latter to more general cases. In this paper, we propose two

[⋆] The full version of this paper, named "Root computation in finite fields", appears in IEICE Trans. Fundamentals, Vol. E96-A, No. 6, pp. 1081 – 1087, 2013, which includes a generalization of the Cipolla-Lehmer method to r-th root cases with r prime.

[⋆⋆] The author is currently with NEC Micro Systems, Ltd.

explicit algorithms for realizing the Cipolla-Lehmer method in the case of cube roots for prime fields $\mathbb{F}_p$ with $p \equiv 1 \pmod 3$. More precisely, we describe two methods for constructing irreducible polynomials of degree three satisfying some conditions to apply the Cipolla-Lehmer method. We implement these methods and compare the results.

The remainder of this paper is organized as follows: In Section 2, we describe the Tonelli-Shanks algorithm and the Cipolla-Lehmer algorithm for square roots computation. In Section 3, we consider the cube root versions of the Tonelli-Shanks method and the Cipolla-Lehmer method, and propose two algorithms of constructing irreducible polynomials of degree three which we need in the Cipolla-Lehmer method. In Section 4, we implement these methods, and compare the results. In Section 5, we give a conclusion and future works.

2 The computation of square roots in finite fields

In this section, we introduce two algorithms of computing square roots in finite fields, that is, the Tonelli-Shanks algorithm [9, 12] and the Cipolla-Lehmer algorithm [4, 6].

Let p be a prime and q a power of p, and $\mathbb{F}_q$ denote a finite field with q elements.

In the case of $p = 2$, there exists a unique square root for each a in $\mathbb{F}_q$, which can be computed easily as $a^{\frac{q}{2}}$. So we assume that p is odd.

2.1 Tonelli-Shanks algorithm

The Tonelli-Shanks algorithm is described in Table 1 ([9, 12], or [7]):

Since this algorithm searches a quadratic non-residue (Step 2), it is in a class of randomized algorithms. Its complexity is $O(v_2(q-1)(\log q)^3)$, where $v_2(q-1)$ denotes the maximum non-negative integer e such that $2^e|(q-1)$. Although the worst time complexity is $O((\log q)^4)$, some improvements with complexities $O((\log q)^3)$ have been proposed in the case where $v_2(q-1)$ is small ([2, 8], or [7]).

2.2 Cipolla-Lehmer algorithm

The Cipolla-Lehmer algorithm is described in Table 2 ([4, 6], or [7]):

From the same reason as the Tonelli-Shanks algorithm, the Cipolla-Lehmer algorithm is also in a class of randomized algorithms. Its complexity is $O((\log q)^3)$, which does not depend on the exponent of 2 in $q-1$, unlike the Tonelli-Shanks algorithm.

Table 1. The Tonelli-Shanks algorithm

```
 Input: A quadratic residue $a$ in $\mathbb{F}_q$
        with odd characteristic.
Output: A square root of $a$.

Step 1: Compute the integers $s$, $t$
        such that $q - 1 = 2^s t$ with $t$ odd.
Step 2: Select a quadratic non-residue $b$ in $\mathbb{F}_q$.
        Compute $c = b^t$ and $r = a^t$.
Step 3: (Computation of the square root of $(a^t)^{-1}$, say $h$.)
        $h \leftarrow 1$, $c \leftarrow c^{-1}$
        for $i = 1$ to $s - 1$
          Compute $d = r^{2^{s-i-1}}$.
          if $d \neq 1$, then $h \leftarrow h \cdot c$, $r \leftarrow r \cdot c^2$.
          $c \leftarrow c^2$.
        end for
Step 4: Return $a^{\frac{t+1}{2}} \cdot h$.
```

Table 2. The Cipolla-Lehmer algorithm

```
 Input: A quadratic residue $a$ in $\mathbb{F}_q$
        with odd characteristic.
Output: A square root of $a$.

Step 1: Select an element $b$ such that
        $b^2 - 4a$ is a quadratic non-residue in $\mathbb{F}_q$.
Step 2: Let $f(x)$ be the polynomial $x^2 - bx + a$
Step 3: Return $x^{(q+1)/2} \pmod{f(x)}$.
```

3 The computation of cube roots in finite fields

In this section, we consider the computation of cube roots in prime fields $\mathbb{F}_p$ with $p \equiv 1 \pmod 3$. We assume $p \equiv 1 \pmod{3^2}$ because efficient methods are proposed for the other cases [3]. All algorithms which we describe in this section are in a class of randomized algorithms because we need a cubic non-residue in $\mathbb{F}_p$. We remark that a non-zero element a in $\mathbb{F}_p$ is a cubic non-residue if and only if $a^{\frac{p-1}{3}} \not\equiv 1 \pmod p$, and in that case, $a^{\frac{p-1}{3}}$ is a primitive cube root of unity in $\mathbb{F}_p$.

3.1 The Tonelli-Shanks based algorithm

The Tonelli-Shanks method for square roots computation was extended to the general r-th roots computation by Adleman, Manders and Miller [1]. We describe the algorithm for the case of cube roots in Table 3.

Table 3. The cube root computation based on the Tonelli-Shanks algorithm

```
 Input: A cubic residue $a$ in $\mathbb{F}_p$
Output: A cube root of $a$.

Step 1: Compute the integers $s$, $t$
        such that $p-1 = 3^s t$ with $t = 3k \pm 1$ .
Step 2: Select a cubic non-residue $b$ in $\mathbb{F}_p$.
        Compute $c = b^t$ and $r = a^t$.
Step 3: (Computation of the cube root of $(a^t)^{-1}$, say $h$.)
        $h \leftarrow 1$, $c' \leftarrow c^{3^{s-1}}$, $c \leftarrow c^{-1}$.
        for $i = 1$ to $s-1$
          Compute $d = r^{3^{s-i-1}}$.
          if $d = c'$, then $h \leftarrow h \cdot c$, $r \leftarrow r \cdot c^3$.
          else if $d \neq 1$ (equivalently, $d = c'^2$),
            then $h \leftarrow h \cdot c^2$, $r \leftarrow r \cdot (c^3)^2$.
          $c \leftarrow c^3$.
        end for
Step 4: $r \leftarrow a^k \cdot h$.
        if $t = 3k+1$, then $r \leftarrow r^{-1}$.
        Return $r$.
```

As in the case of quadratic roots, the complexity of the algorithm (Table 3) is depend on the value $v_3(p-1)$ as well as on the size of $\log p$.

3.2 The Cipolla-Lehmer based algorithm

To extend the Cipolla-Lehmer method to the cube roots computation, we need irreducible polynomials of degree three satisfying some conditions. More precisely, given a cubic residue a in $\mathbb{F}_p$, we have to construct an irreducible monic polynomial $f(x)$ of degree three with coefficients in $\mathbb{F}_p$ and constant term $-a$. Once we find such an $f(x)$, we can compute a cube root of a as $x^{\frac{p^2+p+1}{3}} \pmod{f(x)}$ because we have $f(x) = (x-\alpha)(x-\alpha^p)(x-\alpha^{p^2})$ for a root $\alpha \in \mathbb{F}_{p^3}$ of $f(x)$.

Let $f(x)$ be a polynomial of degree three with coefficients in $\mathbb{F}_p$. Then we have

$$f(x)\text{: irreducible} \Longleftrightarrow f(x) \text{ has no factor of degree one.} \tag{1}$$

Since $\prod_{a\in\mathbb{F}_p}(x-a) = x^p - x$, we also have

$$f(x)\text{: irreducible} \Longleftrightarrow \gcd(f(x),\, x^p - x) = 1. \tag{2}$$

We call the search of irreducible polynomials of degree three based on the fact (2) "*the original method*". In the original method, we choose a random polynomial $f(x)$ in the form $x^3 + \beta x - a$ with $\beta \in \mathbb{F}_p$ and see whether $\gcd(f(x),\, x^p - x) = 1$.

In the remainder of this section, we describe two improvements for finding irreducible polynomials of degree three with constant term $-a$.

Dickson based method: L. E. Dickson described a necessary and sufficient condition for a cubic polynomial over finite fields of odd characteristic to be irreducible, which is as follows:

Theorem 1 ([5]). *Let $\mathbb{F}_q$ be a finite field of characteristic $\neq 2, 3$, and $f(x) = x^3 + ax + b$ a polynomial over $\mathbb{F}_q$ of degree three. Then $f(x)$ is irreducible if and only if the following two conditions hold:*

$$\begin{cases} \text{(i) } D(f) \textit{ is a quadratic residue in } \mathbb{F}_q, \textit{ say } D(f) = (3^2\mu)^2 \\ \text{(ii) } \alpha := \frac{1}{2}(-b + \mu\sqrt{-3}) \textit{ is a cubic non-residue in } \mathbb{F}_q(\sqrt{-3}), \end{cases}$$

where $D(f) = -(4a^3 + 27b^2)$ denotes the discriminant of $f(x)$. We see that the element α is a root of the quadratic equation $x^2 + bx - 3^{-3}a^3 = 0$.

Since $p \equiv 1 \pmod 3$ in our case, we see that $(\frac{-3}{p}) = (\frac{p}{3}) = 1$ by the quadratic reciprocity law, where $(\frac{\cdot}{\cdot})$ denotes the Legendre symbol. Therefore, we have $\mathbb{F}_p(\sqrt{-3}) = \mathbb{F}_p$.

Based on Theorem 1, we describe an algorithm which we call "*Proposed method 1*", for constructing a polynomial of degree three with constant term $-a$ (Table 4).

Linear transformation based method: We propose another method to search irreducible polynomials over $\mathbb{F}_p$ of degree three, which is based on the following fact:

$$f(x)\text{: irreducible} \Longleftrightarrow f(x+\alpha)\text{: irreducible for any } \alpha \text{ in } \mathbb{F}_p. \tag{3}$$

Table 4. Proposed algorithm 1

Input:	A cubic residue a in $\mathbb{F}_p$.
Output:	An irreducible monic polynomial with constant term $-a$.
Step 1:	Choose an element β in $\mathbb{F}_p$ at random. $f(x) \longleftarrow x^3 + \beta x - a$. $D(f) \longleftarrow -(4\beta^3 + 27a^2)$.
Step 2:	**if** $D(f)$ is zero or a quadratic non-residue, then go to Step 1. **else** $\alpha \longleftarrow \frac{1}{2}(a + 3^{-2}\sqrt{-3D(f)})$ (i.e, α: a root of $x^2 - ax - 3^{-3}\beta^3 = 0$).
Step 3:	**if** α is a cubic non-residue, then return $f(x)$. **else** go to Step 1.

We now consider the polynomial in the form $f(x) = x^3 - \alpha$ over $\mathbb{F}_p$. If α is a cubic non-residue, that is, if there exists no element β in $\mathbb{F}_p$ such that $\beta^3 = \alpha$, then $f(x)$ is irreducible from the fact (1).

Combining this with the fact (3), we propose an algorithm which we call "*Proposed method 2*", for constructing a polynomial of degree three with constant term $-a$ (Table 5). We remark that, for Steps 1 and 3, the number of β's in $\mathbb{F}_p$ such that $a+\beta^3$ is a cubic non-residue is approximately $\frac{2}{3}p$ for all sufficiently large p (see Appendix).

Table 5. Proposed algorithm 2

Input:	A cubic residue a in $\mathbb{F}_p$.
Output:	An irreducible monic polynomial with constant term $-a$ or a cube root of a.
Step 1:	Choose an element β in $\mathbb{F}_p$ at random. $\alpha \longleftarrow a + \beta^3$.
Step 2:	**if** $\alpha = 0$, then return $-\beta$ as a cube root of a.
Step 3:	**if** $\alpha^{\frac{p-1}{3}} \neq 1$, then return $(x+\beta)^3 - \alpha$ as an irreducible polynomial. **else** go to Step 1.

4 Experimental results

In this section, we implement the algorithms for computing cube roots based on the proposed methods for various cases, which are listed in Tables 6 – 9. We compute 100 times for each case and list the average time. For the quadratic roots computation in Step 2 of "Proposed method 1" (Table 4), we use the Tonelli-Shanks method (Table 1). This implies that the running time of "Proposed method 1" depends on the value $v_2(p-1)$ as well as on the size of $\log p$.

All computations are performed on a 3 GHz Pentium IV with 0.99 Gb RAM. The language is C with Borland C++ compiler 5.5.1 and with no mathematical library.

If the value $v_3(p-1)$ is not so large, the Tonelli-Shanks method is more efficient than the Cipolla-Lehmer method. But, if the value $v_3(p-1)$ is large, the Cipolla-Lehmer method is more efficient.

For the Cipolla-Lehmer method, we see that the running time is not depend on the value $v_3(p-1)$ and that the proposed methods 1 and 2 are about twice faster than the original method. We further observe the following: (i) In the step of finding an irreducible polynomial $f(x)$ of degree three with a given constant term, the proposed method 2 is more efficient than the proposed method 1. In our implementations, the average number of trials for finding $f(x)$ in the proposed method 1 (resp. the proposed method 2) is about 3 (resp. about 1.5) for a cube root computation. (ii) For the computation of $x^{\frac{p^2+p+1}{3}} \pmod{f(x)}$, the proposed method 1 is more efficient than the proposed method 2 because $f(x)$ is a trinomial for the proposed method 1, while $f(x)$ is a quadrinomial for the proposed method 2.

Table 6. Running time (ms) for $\mathbb{F}_p$ with the size of p being 250 bits

the value of $v_3(p-1)$		2	10	20	30	40	50	60	70	80	90	100
Tonelli-Shanks		3.9	3.4	4.5	7.7	7.2	9.2	12.2	15.1	19.2	23.1	29.2
Cipolla -Lehmer	Original	28.4	32.2	18.6	19.4	20.9	19.8	21.9	20.0	21.2	20.5	19.1
	Proposed 1	18.0	17.2	12.2	10.3	13.0	12.7	10.5	12.2	12.6	10.6	10.6
	Proposed 2	13.6	10.9	10.6	10.3	10.9	10.8	10.8	10.9	10.9	10.9	10.9

Table 7. Running time (ms) for $\mathbb{F}_p$ with the size of p being 500 bits

the value of $v_3(p-1)$		2	20	40	60	80	100	120	140	160	180	200
Tonelli-Shanks		34.5	32.5	35.6	47.8	76.7	102.5	123.4	167.7	207.3	266.9	314.1
Cipolla -Lehmer	Original	132.8	140.2	120.1	121.9	125.6	121.9	121.6	120.9	121.6	130.6	127.2
	Proposed 1	65.1	77.5	63.6	75.0	78.1	73.7	78.6	74.1	64.4	77.3	74.4
	Proposed 2	65.9	66.4	65.5	63.0	65.0	65.0	64.4	65.3	64.4	66.1	64.1

Table 8. Running time (ms) for $\mathbb{F}_p$ with the size of p being 750 bits

the value of $v_3(p-1)$		2	30	60	90	120	150	180	210	240	270	300
Tonelli-Shanks		68.0	75.2	113.6	175.3	269.1	400.3	528.3	720.2	883.1	1142.3	1371.7
Cipolla -Lehmer	Original	387.5	361.4	404.4	337.2	347.2	361.1	344.5	371.2	363.1	364.5	330.8
	Proposed 1	191.1	216.4	238.4	183.0	185.3	230.5	185.3	220.6	221.7	226.6	217.6
	Proposed 2	186.9	186.1	194.1	190.8	189.4	193.9	187.0	187.7	185.9	190.9	188.1

5 Conclusion and future works

In this paper, we considered the computation of cube roots in finite fields based on the Tonelli-Shanks method and the Cipolla-Lehmer one, and described the computational procedure for realizing the Cipolla-Lehmer method in the case of cube roots. More precisely, we proposed two methods for constructing irreducible polynomials of degree three. We further implemented these methods and compare them.

We leave the detailed analysis of these methods, for example the theme whether there exists a deterministic algorithm for cube roots computation with(out) the Genelized Riemann Hypothesis, and the extension of the Cipolla-Lehmer method to the general r-th roots computation. These are our future works.

References

1. L. Adleman, K. Manders and G. Miller, *On taking roots in finite fields*, Proc. 18th IEEE Symposium on Foundations on Computer Science (FOCS), pp. 175 – 177, 1977.
2. A. O. L. Atkin, *Probabilistic primality testing*, Summary by F. Morain, INRIA Res. Rep. 1779, pp. 159 – 163, 1992.
3. P. S. L. M. Barreto and J. F. Voloch, *Efficient computation of roots in finite fields*, Designs, Codes and Cryptography, **39**, pp. 275 – 280, 2006.

Table 9. Running time (ms) for $\mathbb{F}_p$ with the size of p being 1000 bits

the value of $v_3(p-1)$		2	40	80	120	160	200	240	280	320	360	400
Tonelli-Shanks		152.7	169.2	354.2	487.0	780.0	1125.0	1570.5	2041.2	2691.2	3315.5	3559.1
Cipolla -Lehmer	Original	814.4	919.7	839.7	828.7	774.8	841.2	874.5	811.4	785.0	849.8	847.0
	Proposed 1	416.6	505.5	423.7	431.2	499.1	425.1	521.1	410.3	422.1	500.0	430.3
	Proposed 2	412.2	420.9	431.2	429.5	417.3	418.4	430.2	421.9	428.9	419.4	433.9

4. M. Cipolla, *Un metodo per la risolutione della congruenza di secondo grado*, Rendiconto dell'Accademia Scienze Fisiche e Matematiche, Napoli, Ser. 3, Vol. IX, pp. 154 – 163, 1903.
5. L. E. Dickson, *Criteria for the irreducibility of functions in a finite field*, Bull. Amer. Math. Soc., Vol. **13**, No. 1, pp. 1–8, 1906.
6. D. H. Lehmer, *Computer technology applied to the theory of numbers*, Studies in Number Theory, Englewood Cliffs, NJ: Pretice-Hall, pp. 117 – 151, 1969.
7. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.
8. S. Müller, *On the computation of square roots in finite fields*, Designs, Codes and Cryptography, **31**, pp. 301 – 312, 2004.
9. D. Shanks, *Five number-theoretic algorithms*, Proc. 2nd Manitoba Conf. Numer. Math., Manitoba, Canada, pp. 51 – 70, 1972.
10. J. H, Silverman and J. Tate, *Rational Points on Elliptic Curves*, Undergraduate Texts in Math., vol. **106**, Springer-Verlag, New York, 1992.
11. H. Stichtenoth, *Algebraic Function Fields and Codes*, Springer Universitext, Springer-Verlag, Berlin Heidelberg, 1993.
12. A. Tonelli, *Bemerkung über die Auflösung quadratischer Congruenzen*, Göttinger Nachrichten, pp. 344 – 346, 1891.

Appendix

Unless explicitly stated otherwise, we assume $p \equiv 1 \pmod 3$. In this appendix, for a cubic residue a in $\mathbb{F}_p$, we consider the number

$$N := \#\{x \in \mathbb{F}_p \mid x^3 + a \text{ is a cubic non-residue}\}.$$

Let $a = g^3$ with $g \in \mathbb{F}_p$. Since $x^3 + a = g^3\{(xg^{-1})^3 + 1\}$, we consider the number of $\mathbb{F}_p$-rational points $C(\mathbb{F}_p)$ on the Fermat curve

$$C/\mathbb{F}_p : \quad x^3 + 1 = y^3.$$

Setting $x = \frac{X}{Z}$ and $y = \frac{Y}{Z}$, we see that $C(\mathbb{F}_p)$ contains three points at infinity $[X : Y : Z] = [1 : 1 : 0], [1 : \omega : 0], [1 : \omega^2 : 0]$, where ω

is a primitive cube root of unity. For x in $\mathbb{F}_p$, the number of y's in $\mathbb{F}_p$ satisfying $x^3 + 1 = y^3$ is as follows:

$$\begin{cases} 0 & (x^3 + 1\text{: cubic non-residue}) \\ 1 & (x^3 + 1 = 0\text{, i.e. } x = -1,\ -\omega,\ -\omega^2) \\ 3 & (x^3 + 1\text{: cubic residue}) \end{cases}$$

So we obtain

$$N = p - \{3 + (\#C(\mathbb{F}_p) - 6)/3\} = (p - 1) - \#C(\mathbb{F}_p)/3.$$

Since the Fermat curve $C/\mathbb{F}_p$ has genus one in the case of $p \neq 3$ [11], we have

$$p + 1 - 2\sqrt{p} \leq \#C(\mathbb{F}_p) \leq p + 1 + 2\sqrt{p}$$

from the Hasse-Weil theorem [11]. Therefore, the number $N = (p - 1) - \#C(\mathbb{F}_p)/3$ is approximately $\frac{2}{3}p$ for all sufficiently large p.

Finally, we describe the explicit estimation on the number of $\mathbb{F}_p$-rational points on the projective curve defined by $X^3 + Y^3 + Z^3 = 0$, which comes from Gauss' Disquisitiones Arithmeticae, Article 358:

Theorem 2 ([10]). *We assume $p \neq 3$. Let $C/\mathbb{F}_p$ be the Fermat curve defined by $X^3 + Y^3 + Z^3 = 0$. We have*

$$\#C(\mathbb{F}_p) = \begin{cases} p + 1 & (p \not\equiv 1 \pmod 3) \\ p + 1 + A & (p \equiv 1 \pmod 3), \end{cases}$$

where the rational integer A is chosen so that $4p = A^2 + 27B^2$ with a rational integer B and $A \equiv 1 \pmod 3$.