On the Bounded Sum-of-digits Discrete Logarithm Problem in Finite Fields*

Qi Cheng

School of Computer Science
The University of Oklahoma
Norman, OK 73019, USA
Email: qcheng@cs.ou.edu.

Abstract. In this paper, we study the bounded sum-of-digits discrete logarithm problem in finite fields. Our results concern primarily with fields $\mathbf{F}_{q^n}$ where $n|q-1$. The fields are called Kummer extensions of $\mathbf{F}_q$. It is known that we can efficiently construct an element g with order greater than 2^n in the fields. Let $S_q(\bullet)$ be the function from integers to the sum of digits in their q-ary expansions. We first present an algorithm that given g^e $(0 \leq e < q^n)$ finds e in random polynomial time, provided that $S_q(e) < n$. We then show that the problem is solvable in random polynomial time for most of the exponent e with $S_q(e) < 1.32n$, by exploring an interesting connection between the discrete logarithm problem and the problem of list decoding of Reed-Solomon codes, and applying the Guruswami-Sudan algorithm. As a side result, we obtain a sharper lower bound on the number of congruent polynomials generated by linear factors than the one based on Stothers-Mason ABC-theorem. We also prove that in the field $\mathbf{F}_{q^{q-1}}$, the bounded sum-of-digits discrete logarithm with respect to g can be computed in random time $O(f(w)\log^4(q^{q-1}))$, where f is a subexponential function and w is the bound on the q-ary sum-of-digits of the exponent, hence the problem is fixed parameter tractable. These results are shown to be generalized to Artin-Schreier extension $\mathbf{F}_{p^p}$ where p is a prime. Since every finite field has an extension of reasonable degree which is a Kummer extension, our result reveals an unexpected property of the discrete logarithm problem, namely, the bounded sum-of-digits discrete logarithm problem in any given finite field becomes polynomial time solvable in certain low degree extensions.

1 Introduction and Motivations

Most of practical public key cryptosystems base their security on the hardness of solving the integer factorization problem or the discrete logarithm problem in finite fields. Both of the problems admit subexponential algorithms, thus we have to use long parameters, which make the encryption/decryption costly if the parameters are randomly chosen. Parameters of low Hamming weight, or

* This research is partially supported by NSF Career Award CCR-0237845.

more generally, of small sum-of-digits, offer some remedy. Using them speeds up the system while seeming to keep the security intact. In particular, in the cryptosystem based on the discrete logarithm problem in finite fields of small characteristic, using small sum-of-digits exponents is very attractive, due to the existence of normal bases [1]. It is proposed and implemented for smart cards and mobile devices, where the computing power is severely limited. Although attacks exploring the specialty were proposed [14], none of them have polynomial time complexity.

Let $\mathbf{F}_{q^n}$ be a finite field. For $\beta \in \mathbf{F}_{q^n}$, if $\beta, \beta^q, \beta^{q^2}, \cdots, \beta^{q^{n-1}}$ form a linear basis of $\mathbf{F}_{q^n}$ over $\mathbf{F}_q$, we call them *a normal basis*. It is known that a normal basis exists for every pair of prime power q and a positive integer n [11, Page 29]. Every element α in $\mathbf{F}_{q^n}$ can be represented as

$$\alpha = a_0\beta + a_1\beta^q + \cdots + a_{n-1}\beta^{q^{n-1}}$$

where $a_i \in \mathbf{F}_q$ for $0 \leq i \leq n-1$. The power of q is a linear operation, thus

$$\alpha^q = a_0\beta^q + \cdots + a_{n-2}\beta^{q^{n-1}} + a_{n-1}\beta.$$

Hence to compute the q-th power, we only need to shift the digits, which can be done very fast, possibly on the hardware level. Let e be an integer with q-ary expansion

$$e = e_0 + e_1q + e_2q^2 + \cdots + e_{n-1}q^{n-1} \quad (0 \leq e_i < q \text{ for } 0 \leq i \leq n-1). \quad (1)$$

The sum-of-digits of e in the q-ary expansion is defined as $S_q(e) = \sum_{i=0}^{n-1} e_i$. When $q = 2$, the sum-of-digits becomes the famous Hamming weight. To compute α^e, we only need to do shiftings and at most $S_q(e)$ many of multiplications. Furthermore, the exponentiation algorithm can be parallelized, which is a property not enjoyed by the large characteristic fields. For details, see [16].

1.1 Related Work

The discrete logarithm problem in finite field $\mathbf{F}_{q^n}$, is to compute an integer e such that $g' = g^e$, given a generator g of a subgroup of $\mathbf{F}_{q^n}^*$ and g' in the subgroup. The general purpose algorithms to solve the discrete logarithm problem are the number field sieve and the function field sieve (for a survey see [13]). They have time complexity

$$\exp(c(\log q^n)^{1/3}(\log\log q^n)^{2/3})$$

for some constant c, when q is small, or n is small.

Suppose we want to compute the discrete logarithm of g^e with respect to base g in the finite field $\mathbf{F}_{q^n}$. If we know that the Hamming weight of e is equal to w, there is an algorithm proposed by Coppersmith (described in [14]), which works well if w is very small. It is a clever adaption of the baby-step giant-step idea, and runs in random time $O(\sqrt{w}\binom{\lfloor \log q^n/2 \rfloor}{\lfloor w/2 \rfloor})$. It is proved in [14] that the average-case complexity achieves only a constant factor speed-up over the worst case. It is not clear how his idea can be generalized when the exponent has small sum-of-digits in the base $q > 2$. However, we can consider the very special case where $e_i \in \{0,1\}$ for $0 \leq i \leq n-1$ and $\sum_{0 \leq i \leq n-1} e_i = \lfloor \frac{n}{2} \rfloor$. Recall that e_i's are the digits of e in the q-ary expansion. It can be verified that Coppersmith's algorithm can be applied in this case. The time complexity becomes $O(\sqrt{n}\binom{\lfloor n/2 \rfloor}{\lfloor n/4 \rfloor})$. If $q < n^{O(1)}$, it is much worse than the time complexity of the function field sieve on a general exponent.

If the q-ary sum-of-digits of the exponent is bounded by w, is there an algorithm which runs in time $f(w)\log^c(q^n)$ and solves the discrete logarithm problem in $\mathbf{F}_{q^n}$, for some function f and a constant c? A similar problem has been raised from the parametric point of view by Fellows and Koblitz [10], where they consider the prime finite fields and the bounded Hamming weight exponents. Their problem is listed among the most important open problems in the theory of parameterized complexity [9]. From the above discussions, it is certainly more relevant to cryptography to treat the finite fields with small characteristic and exponents with bounded sum-of-digits.

Unlike the case of the integer factorization, where a lot of special purpose algorithms exist, the discrete logarithm problem is considered more intractable in general. As an example, one should not use a RSA modulus of about 1000 bits with one prime factor of 160 bits. It would be vulnerable to the elliptic curve factorization algorithm. However, in the Digital Signature Standard, adopted by the U.S. government, the finite field has cardinality about 2^{1024} or larger, while the encryption/decryption is done in a subgroup of cardinality about 2^{160}. As another example, one should search for a secret prime as random as possible in RSA, while in the case of the discrete logarithm problem, one may use a finite field of small characteristic, hence the group of very special order. It is believed that no trapdoor can be placed in the group order, as long as it has a large prime factor (see the panel report on this issue in the Proceeding of Eurocrypt 1992). In order to have an efficient algorithm to solve the discrete logarithm, we need that every prime factor of the group order is bounded by a polynomial function on the logarithm of the cardinality of the field. Given the current state of analytic number theory, it is very hard, if not impossible, to decide whether there exists *infinitely* many finite fields of even (or constant) characteristic, where the discrete logarithm can be solved in polynomial time.

In summary, there are several common perceptions about the discrete logarithm problem in finite fields:

1. As long as the group order has a big prime factor, the discrete logarithm problem is hard. We may use exponents with small sum-of-digits, since the discrete logarithm problem in that case seems to be fixed parameter intractable. We gain advantage in speed by using bounded sum-of-digits exponents, and at the same time keep the problem as infeasible as using the general exponents.
2. If computing discrete logarithm is difficult, it should be difficult for any generator of the group. The discrete logarithm problem with respect to one generator can be reduced to the discrete logarithm problem with respect

to any generator. Even though in the small sum-of-digits case, a reduction is not available, it is not known that changing the generator of the group affects the hardness of the discrete logarithm problem.

1.2 Our Results

In this paper, we show that those assumptions taken in combination are incorrect. We study the discrete logarithm problem in large multiplicative subgroups of the Kummer and Artin-Schreier extensions with a prescribed base, and prove that the bounded sum-of-digits discrete logarithm are easy in those groups. More precisely we prove constructively:

Theorem 1. *(Main) There exists a random algorithm to find the integer e given g and g^e in $\mathbf{F}_{q^n}$ in time polynomial in $\log(q^n)$ under the conditions:*

1. *$n|q-1$;*
2. *$0 \leq e < q^n$, and $S_q(e) \leq n$;*
3. *$g = \alpha + b$ where $\mathbf{F}_q(\alpha) = \mathbf{F}_{q^n}$, $b \in \mathbf{F}_q^*$ and $\alpha^n \in \mathbf{F}_q$.*

Moreover, there does not exist an integer $e' \neq e$ satisfying that $0 \leq e' < q^n$, $S_q(e') \leq n$ and $g^{e'} = g^e$

The theorem leads directly to a parameterized complexity result concerning the bounded sum-of-digits discrete logarithm, which answers an important open question for special, yet non-negligibly many, cases.

Corollary 1. *There exists an element g of order greater than 2^q in $\mathbf{F}_{q^{q-1}}^*$, such that the discrete logarithm problem with respect to the generator g can be solved in time $f(w)\log^4(q^{q-1})$, where f is a subexponential function and w is the bound of the sum-of-digits of the exponent in q-ary expansion.*

A few comments are in order:

- For a finite field $\mathbf{F}_{q^n}$, if $n|q-1$, then there exists $g \in \mathbf{F}_{q^n}$ satisfying the condition in the theorem, in the other words, there exists an irreducible polynomial of form $x^n - a$ ($a \in \mathbf{F}_q$) over $\mathbf{F}_q$; if there exists α such that $\mathbf{F}_q(\alpha) = \mathbf{F}_{q^n}$ and $\alpha^n \in \mathbf{F}_q$, then $n|q-1$.
- As a comparison, Coppersmith's algorithm runs in exponential time in the case where $e_i \in \{0,1\}$ for $0 \leq i \leq n-1$, $S_q(e) = \frac{n}{2}$ and $q < n^{O(1)}$, while our algorithm runs in polynomial time in that case. On the other hand, Coppersmith's algorithm works for every finite field, while our algorithm works in Kummer extensions. Our result has an indirect affect on an arbitrary finite field though, since every finite field has extensions of degree close to a given number, which are Kummer extensions. As an example, suppose we want to find such an extension of $\mathbf{F}_q$ with degree about $\log^2 q$. We first pick a random n close to $\log q$ such that $(n,q) = 1$. Let l be the order of q in $\mathbf{Z}/n\mathbf{Z}$. The field $\mathbf{F}_{(q^l)^n}$ is a Kummer extension of $\mathbf{F}_{q^l}$, and an extension of $\mathbf{F}_q$. According to Theorem 1, there is a polynomial time algorithm which computes

the discrete logarithm to some element g in $\mathbf{F}_{q^{ln}}$ provided that the sum-of-digits of the exponent in the q^l-ary expansion is less than n. Hence our result reveals an unexpected property of the discrete logarithm problem in finite fields: the difficulty of bounded sum-of-digits discrete logarithm problem drops dramatically if we move up to extensions and increase the base of the exponent accordingly.

- Numerical evidences suggest that the order of g is often equal to the group order $q^n - 1$, and is close to the group order otherwise. However, it seems hard to prove it. In fact, this is one of the main obstacles in improving the efficiency of AKS-style primality testing algorithm [2]. We make the following conjecture.

  Conjecture 1. Suppose that a finite field $\mathbf{F}_{q^n}$ and an element g in the field satisfy the conditions in Theorem 1. In addition, $n \geq \log q$. The order of g is greater than $q^{n/c}$ for an absolute constant c.

- Even though we can not prove that the largest prime factor of the order of g is very big, it seems, as supported by numerical evidences, that the order of g, which is a factor of $q^n - 1$ bigger than 2^n, is rarely smooth. For instance, in the $\mathbf{F}_{2^{889}} = \mathbf{F}_{128^{127}}$, any g generates the whole group $\mathbf{F}^*_{2^{889}}$. The order $2^{889} - 1$ contains a prime factor of 749 bits. One should not attempt to apply the Silver-Pohlig-Hellman algorithm here.

A natural question arises: can the restriction on the sum-of-digits in Theorem 1 be relaxed? Clearly if we can solve the problem under condition $S_q(e) \leq (q-1)n$ in polynomial time, then the discrete logarithm problem in subgroup generated by g is broken. If g is a generator of $\mathbf{F}^*_{q^n}$, then the discrete logarithm problem in $\mathbf{F}_{q^n}$ and any of its subfields to any base are broken. We find a surprising relationship between the relaxed problem and the list decoding problem of Reed-Solomon codes. We are able to prove:

Theorem 2. *Suppose e is chosen randomly from the set*

$$\{0 \leq e < q^n - 1 | S_q(e) < 1.32n\}.$$

There exists an algorithm given g and g^e in $\mathbf{F}_{q^n}$, to find e in time polynomial in $\log(q^n)$, with probability greater than $1 - c^{-n}$ for some constant c greater than 1, under the conditions:

1. *$n | q - 1$;*
2. *$g = \alpha + b$ where $\mathbf{F}_q(\alpha) = \mathbf{F}_{q^n}$, $b \in \mathbf{F}^*_q$ and $\alpha^n \in \mathbf{F}_q$.*

Given a polynomial ring $\mathbf{F}_q[x]/(h(x))$, it is an important problem to determine the size of multiplicative subgroup generated by $x - s_1, x - s_2, \cdots, x - s_n$ where $(s_1, s_2, \cdots, s_n) = S$ is a list of distinct elements in $\mathbf{F}_q$, and for all i, $h(s_i) \neq 0$. The lower bound of the order directly affects the time complexity of AKS-style primality proving algorithm. In that context, we usually have $\deg h(x) | n$. Assume that $\deg h(x) = n$. For a list of integers $E = (e_1, e_2, \cdots, e_n)$, we denote

$$(x - s_1)^{e_1}(x - s_2)^{e_2} \cdots (x - s_n)^{e_n}$$

by $(x-S)^E$. One can estimate the number of distinct congruent polynomials of form $(x-S)^E$ modulo $h(x)$ for E in certain set. It is obvious that if $E \in \{(e_1, e_2, \cdots, e_n) | \sum e_i < n-1, e_i \geq 0\}$, then all the polynomials are in different congruent classes. This gives a lower bound of 4^n. Through a clever use of Stothers-Mason ABC-theorem, Voloch [15] and Berstein [5] proved that if $\sum e_i < 1.1n$, then at most 4 such polynomials can fall in the same congruent class, hence obtained a lower bound of 4.27689^n. We improve their result and obtain a lower bound of 5.17736^n.

Theorem 3. *Use the above notations. Let C be*

$$\{(e_1, e_2, \cdots, e_n) | e_i \geq 0 \text{ for } 1 \leq i \leq n, \sum_{i=1}^{n} e_i < 1.5501n, |\{i | e_i \neq 0\}| = \lfloor 0.7416n \rfloor\}.$$

If there exist pairwise different element $E_1, E_2, \cdots, E_m \in C$ such that

$$(x-S)^{E_1} \equiv (x-S)^{E_2} \equiv \cdots \equiv (x-S)^{E_m} \pmod{h(x)},$$

then $m = O(n^2)$. Note that $|C| = 5.17736^n n^{\Theta(1)}$

By allowing negative exponents, Voloch [15] obtained a bound of 5.828^n. Our bound is smaller than his. However, starting from $|S| = 2\deg h(x)$, our method gives better bounds. Details are left in the full paper. A distinct feature of our bound is that it relates to the list decoding algorithm of Reed-Solomon codes. If a better list decoding algorithm is found, then our bound can be improved accordingly.

1.3 Organization of the Paper

The paper is organized as follows. In Section 2, we list some results of counting numbers with small sum-of-digits. In Section 3, we present the basic idea and the algorithm, and prove Theorem 1 and Corollary 1. In Section 4, we prove Theorem 2 and Theorem 3. In Section 5, we extend the results to Artin-Schreier extensions. We conclude our paper with discussions of open problems.

2 Numbers with Small Sum-of-digits

Suppose that the q-ary expansion of a positive integer e is

$$e = e_0 + e_1 q + e_2 q^2 + \cdots + e_{n-1} q^{n-1},$$

where $0 \leq e_i \leq q-1$ for all $0 \leq i \leq n-1$. How many nonnegative integers e less than q^n satisfy $S_q(e) = w$? Denote the number by $N(w, n, q)$. Then $N(w, n, q)$ equals the number of nonnegative integral solutions of

$$\sum_{i=0}^{n-1} e_i = w$$

under the conditions that $0 \le e_i \le q-1$ for all $0 \le i \le n-1$. The generating function for $N(w,n,q)$ is

$$(1+x+\cdots+x^{q-1})^n = \sum_i N(i,n,q)x^i.$$

If $w \le q-1$, then the conditions $e_i \le q-1$ can be removed, we have that $N(w,n,q) = \binom{w+n-1}{n-1}$. It is easy to see that if $q=2$, we have that $N(w,n,2) = \binom{n}{w}$. In the later section, we will need to estimate $N(w,n,q)$, where w is n times a small constant less than 2. Since

$$\begin{aligned}
&(1+x+\cdots+x^{q-1})^n \\
&= (\frac{1-x^q}{1-x})^n \\
&= (1-x^q)^n \sum_{i=0}^{\infty} \binom{i+n-1}{n-1} x^i \\
&\equiv (1-nx^q) \sum_{i=0}^{2q-1} \binom{i+n-1}{n-1} x^i \pmod{x^{2q}} \\
&\equiv \sum_{i=0}^{q-1} \binom{i+n-1}{n-1} x^i + \sum_{i=q}^{2q-1} (\binom{i+n-1}{n-1} - n\binom{i-q+n-1}{n-1}) x^i \pmod{x^{2q}}
\end{aligned}$$

Hence $N(w,n,q) = \binom{w+n-1}{n-1} - n\binom{w-q+n-1}{n-1}$ if $q \le w < 2q$.

3 The Basic Ideas and the Algorithm

The basic idea of our algorithm is adopted from the index calculus algorithm. Let $\mathbf{F}_{q^n}$ be a Kummer extension of $\mathbf{F}_q$, namely, $n|q-1$. Assume that $q = p^d$ where p is the characteristic. The field $\mathbf{F}_{q^n}$ is usually given as $\mathbf{F}_p[x]/(u(x))$ where $u(x)$ is an irreducible polynomial of degree dn over $\mathbf{F}_p$. If g satisfies the condition in Theorem 1, then $x^n - \alpha^n$ must be an irreducible polynomial over $\mathbf{F}_q$. Denote α^n by a. To implement our algorithm, it is necessary that we work in another model of $\mathbf{F}_{q^n}$, namely, $\mathbf{F}_q[x]/(x^n - a)$. Fortunately the isomorphism

$$\psi : \mathbf{F}_p[y]/(u(y)) \rightarrow \mathbf{F}_{q^n} = \mathbf{F}_q[x]/(x^n - a)$$

can be efficiently computed. To compute $\psi(v(y))$, where $v(y)$ is a polynomial of degree at most $dn-1$ over $\mathbf{F}_p$, all we have to do is to factor $u(y)$ over $\mathbf{F}_q[x]/(x^n-a)$, and to evaluate $v(y)$ at one of the roots. Factoring polynomials over finite fields is a well-studied problem in computational number theory, we refer to [3] for a complete survey of results. The random algorithm runs in expected time $O(dn(dn + \log q^n)(dn \log q^n)^2)$, and the deterministic algorithm runs in time $O(dn(dn+q)(dn \log q^n)^2)$. From now on we assume the model $\mathbf{F}_q[x]/(x^n - a)$.

Consider the subgroup generated by $g = \alpha + b$ in $(\mathbf{F}_q[x]/(x^n - a))^*$, recall that $b \in \mathbf{F}_q^*$ and $\alpha = x \pmod{x^n - a}$. The generator g has order greater than 2^n [8], and has a very nice property as follows. Denote $a^{\frac{q-1}{n}}$ by h, we have

$$g^q = (\alpha + b)^q = \alpha^q + b = a^{\frac{q-1}{n}}\alpha + b = h\alpha + b,$$

and more generally

$$(\alpha + b)^{q^i} = \alpha^{q^i} + b = h^i\alpha + b.$$

In other words, we obtain a set of relations: $\log_{\alpha+b}(h^i\alpha + b) = q^i$ for $0 \leq i \leq n-1$. This corresponds to the precomputation stage of the index calculus. The difference is that, in our case, the stage finishes in polynomial time, while generally it requires subexponential time. For a general exponent e,

$$\begin{aligned}(\alpha + b)^e &= (\alpha + b)^{e_0 + e_1 q + \cdots + e_{n-1}q^{n-1}} \\ &= (\alpha + b)^{e_0}(h\alpha + b)^{e_1} \cdots (h^i\alpha + b)^{e_i} \cdots (h^{n-1}\alpha + b)^{e_{n-1}}.\end{aligned}$$

If $f(\alpha)$ is an element in $\mathbf{F}_{q^n}$, where $f \in \mathbf{F}_q[x]$ is a polynomial of degree less than n, and $f(\alpha) = (\alpha + b)^e$ and $S_q(e) < n$, then due to unique factorization in $\mathbf{F}_q[x]$, *$f(x)$ can be completely split into the product of linear factors over $\mathbf{F}_q$. We can read the discrete logarithm from the factorizations, after the coefficients are normalized.* The algorithm is described as follows.

Algorithm 1 *Input: g, g^e in $\mathbf{F}_{q^n} = \mathbf{F}_q[x]/(x^n - a)$ satisfying the conditions in Theorem 1.*

Output: e.

1. *Define an order in $\mathbf{F}_q$ (for example, use the lexicographic order). Compute and sort the list $(1, h, h^2, h^3, \cdots, h^{n-1})$.*
2. *Suppose that g^e is represented by $f(\alpha)$, where $f \in \mathbf{F}_q[x]$ has degree less than n. Factoring $f(x)$ over $\mathbf{F}_q$, let $f(x) = c(x + d_1)^{e_1} \cdots (x + d_k)^{e_k}$ where $c, d_1, \cdots, d_k$ are in $\mathbf{F}_q$.*
3. *(Normalization) Normalize the coefficients and reorder the factors of $f(x)$ such that their constant coefficients are b and $f(x) = (x + b)^{e_1} \cdots (h_{n-1}x + b)^{e_{n-1}}$, where $h_i = h^i$;*
4. *Output $e_0 + e_1 q + \cdots + e_{n-1}q^{n-1}$;*

The step 1 takes time $O(n \log^2 q \log n + n \log n \log q) = O(n \log n \log^2 q)$. The most time-consuming part is to factor a polynomial over $\mathbf{F}_q$ with degree at most n. The random algorithm runs in expected time $O(n(n + \log q)(n \log q)^2)$ and the deterministic algorithm runs in time $O(n(n + q)(n \log q)^2) = O(n^3 q \log^2 q)$. Normalization and reordering can be done in time $O(n \log n \log q)$, since we have a sorted list of $(1, h, h^2, h^3, \cdots, h^{n-1})$. Thus the algorithm can be finished in random time $O(n(n + \log q)(n \log q)^2)$ and in deterministic time $O(n^3 q \log^2 q)$. This concludes the proof of the main theorem.

Now we are ready to prove Corollary 1. Any $f(x)$ where $f(\alpha) = (\alpha + b)^e \in <\alpha + b> \subseteq \mathbf{F}_{q^{q-1}}$ is congruent to a product of at most $w = S_q(e)$ linear factors modulo $x^{q-1} - a$. If $w < q-1$, we have an algorithm running in time $O(q^4 \log^2 q)$, according to Theorem 1. So we only need to consider the case when $w \geq q - 1$. The general purpose algorithm will run in random time $f(\log q^{q-1})$, where f is a subexponential function. Theorem 1 follows from the fact that $\log q^{q-1} \leq w \log w$.

4 The Application of the List Decoding Algorithm of Reed-Solomon Codes

A natural question arises: can we relax the bound on the sum-of-digits and still get a polynomial time algorithm? Solving the problem under the condition $S_q(e) \leq (q-1)n$ basically renders the discrete logarithm problems in $\mathbf{F}_{q^n}$ and any of its subfields easy. Suppose that $g^e = f(\alpha)$ where $f(x) \in \mathbf{F}_q[x]$ has degree less than n. Using the same notations as in the previous section, we have

$$f(\alpha) = (\alpha + b)^{e_0}(h\alpha + b)^{e_2} \cdots (h^{n-1}\alpha + b)^{e_{n-1}}.$$

Hence there exists a polynomial $t(x)$ with degree $\sum_{i=0}^{n-1} e_i - n$ such that

$$f(x) + (x^n - a)t(x) = (x + b)^{e_0}(hx + b)^{e_1} \cdots (h^{n-1}x + b)^{e_{n-1}}.$$

If the cardinality of $\{i | e_i \neq 0\}$ is greater than k then the curve $y = t(x)$ will pass at least k points in the set

$$\{(i, -\frac{f(i)}{i^{q-1} - a}) | i \in \{-b, -\frac{b}{h}, \cdots, -\frac{b}{h^{n-1}}\}\}.$$

To find all the polynomials of degree $d = \sum_{i=0}^{n-1} e_i - n$, which pass at least k points in a given set of n points, is an instance of the list decoding problem of Reed-Solomon codes. It turns out that there are only a few of such polynomials, and they can be found efficiently as long as $k \geq \sqrt{nd}$.

Proposition 1. *(Guruswami-Sudan [12]) Given n distinct elements $x_0, x_1, \cdots, x_{n-1} \in \mathbf{F}_q$, n values $y_0, y_1, \cdots, y_{n-1} \in \mathbf{F}_q$ and a natural number d, there are at most $O(\sqrt{n^3 d})$ many univariate polynomials $t(x) \in \mathbf{F}_q[x]$ of degree at most d such that $y_i = t(x_i)$ for at least $\sqrt{nd}$ many points. Moreover, these polynomials can be found in random polynomial time.*

For each $t(x)$, we use the Cantor-Zassenhaus algorithm to factor $f(x) + (x^n - a)t(x)$. There must exist a $t(x)$ such that the polynomial $f(x) + (x^n - a) * t(x)$ can be completely factored into a product of linear factors in $\{h^i x + b | 0 \leq i \leq n-1\}$, and e is computed as a consequence.

4.1 The Proof of Theorem 2

In this section, we consider the case when $S_q(e) \le 1.32n$. If there are at least $0.5657n \ge \sqrt{0.32n \cdot n}$ number of nonzero e_i's, then we can apply the Guruswami-Sudan algorithm to find all the $t(x)$. In order to prove Theorem 2, it remains to show:

Lemma 1. *Define* $A_{n,q}$ *as*

$$\{(e_1, e_2, \cdots, e_n) \mid e_1+e_2+\cdots+e_n \le 1.32n, e_i \in \mathbf{Z} \text{ and } 0 \le e_i \le q-1 \text{ for } 1 \le i \le n.\}$$

and B_n *as*

$$\{(e_1, e_2, \cdots, e_n) \mid |\{i|e_i \neq 0\}| < 0.5657n\}.$$

We have

$$\frac{|A_{n,q} \cap B_n|}{|A_{n,q}|} < c^{-n}$$

for some constant $c > 1$ *when* n *is sufficiently large.*

Proof. The cardinality of $A_{n,q}$ is $\sum_{i=0}^{\lfloor 1.32n \rfloor} N(i,n,q) > \binom{2.32n}{n} > 4.883987...^n$. The cardinality of $A_{n,q} \cap B_n$ is less than $\sum_{v=\lceil 0.5657n \rceil}^{n} \binom{n}{v}\binom{1.32n}{n-v-1}$. The summands maximize at $v = 0.5657n$ if $v \ge 0.5657n$. Hence we have

$$\begin{aligned} &\sum_{v=\lceil 0.5657n \rceil}^{n} \binom{n}{v}\binom{\lfloor 1.32n \rfloor}{n-v-1} \\ &< 0.4343n \binom{n}{\lceil 0.5657n \rceil}\binom{\lfloor 1.32n \rfloor}{\lfloor 0.4343n \rfloor} \\ &< 4.883799...^n \end{aligned}$$

This proves the lemma with $c = 4.883987.../4.883799... > 1$.

4.2 The Proof of Theorem 3

Proof. Let τ be a positive real number less than 1. Define

$$C_{n,q,\tau} = \{(e_1, e_2, \cdots, e_n) \mid \begin{array}{l} e_1 + e_2 + \cdots + e_n = \lfloor (1+\tau)n \rfloor, e_i \in \mathbf{Z} \\ \text{and } 0 \le e_i \le q-1 \text{ for } 1 \le i \le n \\ \text{and } |\{i|e_i \neq 0\}| = \lfloor \sqrt{\tau} n \rfloor \end{array} \}$$

Given $f(x) \in \mathbf{F}_q[x]$, if there exists $E \in C_{n,q,\tau}$, such that $(x-S)^E \equiv f(x) \pmod{h(x)}$, there must exist a polynomial $t(x)$ such that $(x-S)^E = t(x)h(x) + f(x)$, and $t(x)$ is a solution for the list decoding problem with input $\{(s, -\frac{f(s)}{h(s)})|s \in S\}$. According to Propostion 1, there are at most $O(n^2)$ solutions. Thus the number of congruent classes modulo $h(x)$ that $\{(x-S)^E|E \in C_{n,q,\tau}\}$ has is greater than $\Omega(|C_{n,q,\tau}|/n^2)$. We have

$$|C_{n,q,\tau}| = \binom{n}{\sqrt{\tau}n}\binom{(1+\tau)n}{\sqrt{\tau}n} = n^{\Theta(1)}\left(\frac{(1+\tau)^{1+\tau}}{\tau^{\sqrt{\tau}}(1-\sqrt{\tau})^{1-\sqrt{\tau}}(1+\tau-\sqrt{\tau})^{1+\tau-\sqrt{\tau}}}\right)^n.$$

It takes the maximum value $5.17736...^n$ at $\tau = 0.5501$.

5 Artin-Schreier Extensions

Let p be a prime. The Artin-Schreier extension of a finite field $\mathbf{F}_p$ is $\mathbf{F}_{p^p}$. It is easy to show that $x^p - x - a = 0$ is an irreducible polynomial in $\mathbf{F}_p$ for any $a \in \mathbf{F}_p^*$. So we may take $\mathbf{F}_{p^p} = \mathbf{F}_p[x]/(x^p - x - a)$. Let $\alpha = x \pmod{x^p - x - a}$. For any $b \in \mathbf{F}_p$, we have

$$(\alpha + b)^p = \alpha^p + b = \alpha + b + a,$$

and similarly

$$(\alpha + b)^{p^i} = \alpha^{p^i} + b = \alpha + b + ia.$$

Hence the results for Kummer extensions can be adopted to Artin-Schreier extensions. For the subgroup generated by $\alpha + b$, we have a polynomial algorithm to solve the discrete logarithm if the exponent has p-ary sum-of-digits less than p. Note that b may be 0 in this case.

Theorem 4. *There exists an algorithm to find the integer e given g and g^e in $\mathbf{F}_{p^p}$ in time polynomial in $\log p^p$ under the conditions:*

1. $0 \le e < p^p$*, and* $S_q(e) \le p - 1$*;*
2. $g = \alpha + b$ *where* $\mathbf{F}_p(\alpha) = \mathbf{F}_{p^p}$*,* $b \in \mathbf{F}_p$ *and* $\alpha^p + \alpha \in \mathbf{F}_p^*$*.*

Moreover, there does not exist an integer $e' \neq e$ satisfying that $0 \le e' < p^p$, $S_q(e') \le n$ and $g^{e'} = g^e$.

Theorem 5. *There exists an element g of order greater than 2^p in $\mathbf{F}_{p^p}^*$, such that the discrete logarithm problem with respect to g can be solved in time $O(f(w)(\log p^p)^4)$, where f is a subexponential function and w is the bound of the sum-of-digits of the exponent in the p-ary expansion.*

Theorem 6. *Suppose that $g = \alpha + b$, where $\mathbf{F}_p(\alpha) = \mathbf{F}_{p^p}$, $b \in \mathbf{F}_p$ and $\alpha^p + \alpha \in \mathbf{F}_p^*$. Suppose e is chosen in random from the set*

$$\{0 \le e < q^n - 1 | S_q(e) < 1.32n\}.$$

There exists an algorithm given g and g^e in $\mathbf{F}_{p^p}$, to find e in time polynomial in $\log(p^p)$, with probability greater than $1 - c^{-n}$ for some constant c greater than 1.

6 Concluding Remarks

A novel idea in the celebrated AKS primality testing algorithm, is to construct a subgroup of large cardinality through linear elements in finite fields. The subsequent improvements [6, 7, 4] rely on constructing a single element of large order. It is speculated that these ideas will be useful in attacking the integer factorization problem. In this paper, we show that they do affect the discrete logarithm problem in finite fields. We give an efficient algorithm which computes the bounded sum-of-digits discrete logarithm with respect to prescribed bases in Kummer extensions. We believe that this is more than a result which deals with only special cases, as every finite field has extensions of reasonable degrees which are Kummer extensions. For instance, if we need to compute the discrete logarithm of s in $\mathbf{F}_q$ base g, we can construct a suitable Kummer extention $\mathbf{F}_{q^n}$, and try to solve the discrete logarithms of a and g with respect to a selected base in the extension. This approach is worth studying. Another interesting problems is to further relax the restriction on the sum-of-digits of the exponent. It is also important to prove or disprove Conjecture 1. If that conjecture is true, the AKS-style primality proving can be made compatible or even better than ECPP or the cyclotomic testing in practice.

Acknowledgments We thank Professor Pedro Berrizbeitia for very helpful discussions.

References

1. G. B. Agnew, R. C. Mullin, I. M. Onyszchuk, and S. A. Vanstone. An implementation for a fast public-key cryptosystem. *Journal of Cryptology*, 3:63–79, 1991.
2. M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. http://www.cse.iitk.ac.in/news/primality.pdf, 2002.
3. Eric Bach and Jeffrey Shallit. *Algorithmic Number theory*, volume I. The MIT Press, 1996.
4. D. J. Bernstein. Proving primality in essentially quartic random time. http://cr.yp.to/papers/quartic.pdf, 2003.
5. D. J. Bernstein. Sharper ABC-based bounds for congruent polynomials. http://cr.yp.to/, 2003.
6. Pedro Berrizbeitia. Sharpening "primes is in p" for a large family of numbers. http://lanl.arxiv.org/abs/math.NT/0211334, 2002.
7. Qi Cheng. Primality proving via one round in ECPP and one iteration in AKS. In Dan Boneh, editor, *Proc. of the 23rd Annual International Cryptology Conference (CRYPTO)*, volume 2729 of *Lecture Notes in Computer Science*, Santa Barbara, 2003. Springer-Verlag.
8. Qi Cheng. Constructing finite field extensions with large order elements. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004.
9. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.
10. M. Fellows and N. Koblitz. Fixed-parameter complexity and cryptography. In *Proceedings of the Tenth International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC'93)*, volume 673 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.
11. Shuhong Gao. *Normal Bases over Finite Fields*. PhD thesis, The University of Waterloo, 1993.
12. Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.
13. A. M. Odlyzko. Discrete logarithms: The past and the future. *Designs, Codes, and Cryptography*, 19:129–145, 2000.

14. D. R. Stinson. Some baby-step giant-step algorithms for the low Hamming weight discrete logarithm problem. *Math. Comp.*, 71:379–391, 2002.
15. J. F. Voloch. On some subgroups of the multiplicative group of finite rings. http://www.ma.utexas.edu/users/voloch/preprint.html, 2003.
16. Joachim von zur Gathen. Efficient exponentiation in finite fields. In *Proc. 32nd IEEE Symp. on Foundations of Comp. Science*, 1991.