A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic

Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, Emmanuel Thomé

A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic

Improvements over FFS in small to medium characteristic

Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, Emmanuel Thomé

1 Introduction

The discrete logarithm problem (DLP) was first proposed as a hard problem in cryptography in the seminal article of Diffie and Hellman [DH76]. Since then, together with factorization, it has become one of the two major pillars of public key cryptography. As a consequence, the problem of computing discrete logarithms has attracted a lot of attention. From an exponential algorithm in 1976, the fastest DLP algorithms have been greatly improved during the past 35 years. A first major progress was the realization that the DLP in finite fields can be solved in subexponential time, i.e. $L(1/2)$ where $L_N(\alpha) = \exp\left(O((\log N)^\alpha (\log\log N)^{1-\alpha})\right)$. The next step further reduced this to a heuristic $L(1/3)$ running time in the full range of finite fields, from fixed characteristic finite fields to prime fields [Adl79, Cop84, Gor93, Adl94, JL06, JLSV06].

Recently, practical and theoretical progress have been made [Jou13a, GGMZ13, Jou13b] with an emphasis on small to medium characteristic finite fields and composite degree extensions. The most general and efficient algorithm [Jou13b] gives a complexity of $L(1/4 + o(1))$ when the characteristic is smaller than the square root of the extension degree. Among the ingredients of this approach, we find the use of a very particular representation of the finite field; the use of the so-called *systematic equation*; and the use of algebraic resolution of bilinear polynomial systems in the individual logarithm phase.

In the present work, we present a new discrete logarithm algorithm, in the same vein as in [Jou13b] that uses an asymptotically more efficient descent approach. The main result gives a *quasi-polynomial* heuristic complexity for the DLP in finite field of small characteristic. By quasi-polynomial, we mean a complexity of type $n^{O(\log n)}$ where n is the bit-size of the cardinality of the finite field. Such a complexity is smaller than any $L(\varepsilon)$ for $\epsilon > 0$. It remains super-polynomial in the size of the input, but offers a major asymptotic improvement compared to $L(1/4 + o(1))$.

The key features of our algorithm are the following.

- We keep the field representation and the systematic equations of [Jou13b].
- The algorithmic building blocks are elementary. In particular, we avoid the use of Gröbner basis algorithms.
- The complexity result relies on three key heuristics: the existence of a polynomials representation of the appropriate form; the fact that the smoothness probabilities of some non-uniformly distributed polynomials are similar to the probabilities for uniformly random polynomials of the same degree; and the linear independence of some finite field elements related to the action of $\mathrm{PGL}_2(\mathbb{F}_q)$.

The heuristics are very close to the ones used in [Jou13b]. In addition to the arguments in favor of these heuristics already given in [Jou13b], we performed some experiments to validate them on practical instances.

Although we insist on the case of finite fields of small characteristic, where quasi-polynomial complexity is obtained, our new algorithm improves the complexity discrete logarithm computations in a much larger range of finite fields.

More precisely, in finite fields of the form $\mathbb{F}_{q^k}$, where q grows as $L_{q^k}(\alpha)$, the complexity becomes $L_{q^k}(\alpha + o(1))$. As a consequence, our algorithm is asymptotically faster than the Function Field Sieve algorithm in almost all the range previously covered by this algorithm. Whenever $\alpha < 1/3$, our new algorithm offers the smallest complexity. For the limiting case $L(1/3, c)$, the function field sieve remains more efficient for small values of c, and the number field sieve is better for large values of c (see [JLSV06]).

This article is organized as follows. Section 2 presents the setting for which our algorithm applies. Section 3 states the main result, and discusses how it can be used to design a complete discrete logarithm algorithm. The new algorithm for performing a descent step is presented in Section 4. Heuristic justifications are suggested in Section 5, and the consequences for the asymptotic hardness of several DLP problems is discussed in Section 6. A variant is presented in Section 7.

2 Setting

We start by describing the setting in which our algorithm applies. This is essentially the same as the one in [Jou13b]. The only real difference is the fact that we consider a larger range of finite fields with small to medium characteristic.

Let q be a prime power. We consider a field $\mathbb{K}$ that is a finite field extension of $\mathbb{F}_q$ of the form

$$\mathbb{K} = \mathbb{F}_{q^2}[X]/\varphi(X) \cong \mathbb{F}_{q^{2k}},$$

where $\varphi(X)$ verifies:

- φ is an irreducible polynomial of degree k;
- there exist low degree polynomials $h_0(X)$ and $h_1(X)$ in $\mathbb{F}_{q^2}[X]$ such that $\varphi(X) \,|\, h_1(X)X^q - h_0(X)$.

In our complexity estimates, we let q tend to infinity, but the h_i's have degrees bounded by a constant δ. Note that [Jou13b] gives evidence that choosing $\delta = 2$ should be enough. However, any constant degree δ is enough for our complexity analysis.

Our setting imposes that $k \leq q + \delta$, hence, since δ is constant, we have $k = O(q)$. Furthermore, since the end result is super-polynomial in q, we simplify the exposition by counting at unit time cost any step in the computation with complexity bounded by a polynomial in $\log q$.

3 Main result

Proposition 1. *Under plausible heuristics, there exists an algorithm that takes as input a field $\mathbb{K}$ as above, and a polynomial $P(X)$ of degree $D \leq k$ over $\mathbb{F}_{q^2}$ and returns a linear relation between the discrete logarithm of P in $\mathbb{K}$ and the discrete logarithms of $O(q^2 D)$ polynomials of degree less than $\max(2, \lceil D/2 \rceil)$ in time bounded by a polynomial in q and D.*

The algorithm is given in the next section. We first show how to use this as a building block for a complete discrete logarithm algorithm.

Let P be an element of $\mathbb{K}$ for which we want to compute the discrete logarithm. We can assume that P is represented by a polynomial of degree less than k over $\mathbb{F}_{q^2}$. We start by applying the algorithm of Proposition 1 to P. We obtain a relation of the form

$$\log P = \sum e_i \log P_i,$$

where the sum has at most $\kappa q^2 k$ terms for some constant κ and the P_i's have degree less than $\lceil k/2 \rceil$. Then, we apply recursively the algorithm to the P_i's, thus creating a descent procedure where at each step, a given element P is expressed as a product of elements, whose degree is at most half the degree of P (rounded up) and the arity of the descent tree is in $O(q^2 k)$.

At the end of the process, the logarithm of P is expressed as a linear combination of the logarithms of linear polynomials. From [Jou13b], it is known that these can be computed in polynomial time in q. So we are left with the complexity analysis of the descent process.

Each internal node of the descent tree corresponds to one application of the algorithm of Proposition 1, therefore each internal node has a cost which is bounded by a polynomial in q and k. The total cost of the descent is therefore bounded by the number of nodes in the descent tree times a polynomial in q and k. The depth of the descent tree is in $O(\log k)$. At the i-th depth-level, we have at most $(\kappa q^2 k)^i$ polynomials. Therefore, the number of internal nodes is bounded by $(q^2k)^{O(\log k)}$. Since $k = O(q)$, and since any polynomial in q and k is absorbed in the $O()$ notation in the exponent, we obtain the following result.

Theorem 2. *Assuming the same heuristics as in Proposition 1, any discrete logarithm in $\mathbb{K}$ can be computed in time bounded by*

$$q^{O(\log k)}.$$

Let us express this complexity in terms of the size of the input when k is close to q. The cardinality of the finite field $\mathbb{K}$ is $Q = q^{2k} \approx q^{2q}$, so that the bit-size of the input is $\log Q \approx 2q \log q$. And conversely we have the estimate $q \approx \log Q / \log\log Q$. Therefore, the complexity of our discrete logarithm algorithm is in $(\log Q)^{O(\log\log Q)}$. This is smaller than any expression of the form $L_Q(\alpha)$ for $\alpha > 0$. Hence, although the complexity is still superpolynomial, it is much closer to polynomial time complexity than the classical subexponential complexities which are involved in integer factorization or in previously known discrete logarithm algorithms.

4 Algorithm for one descent step

Notation. Let $\mathcal{P}_q$ be a set of 2×2-matrices that form a set of representatives of the left cosets of $\mathrm{PGL}_2(\mathbb{F}_q)$ in $\mathrm{PGL}_2(\mathbb{F}_{q^2})$. Since the group order of $\mathrm{PGL}_2(\mathbb{F}_q)$ is q^3-q for any q, then the cardinality of $\mathcal{P}_q$ is q^3+q. The cost of computing a set of representatives for $\mathcal{P}_q$ with a naive algorithm is polynomial in q. Furthermore, this computation can be shared between all the application of Proposition 1 during the descent. It can also be shared between different finite fields that share the same value of q, so this cost can be neglected.

Proof of Proposition 1. To decompose the logarithm of $P(X)$, we are going to decompose *simultaneously* all the translates of $P(X)$ by a constant in $\mathbb{F}_{q^2}$. This is done with the *systematic equation* already used in [Jou13b]:

$$X^q - X = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha),$$

where the representatives of $\mathbb{P}^1(\mathbb{F}_q)$ are chosen so that the right-hand side is indeed monic.

To each element $m = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ of $\mathcal{P}_q$, we associate the equation E_m obtained by substituting $m \cdot P = \frac{aP+b}{cP+d}$ in place of X in the systematic equation. Clearing denominators with a multiplication by $(cP+d)^{q+1}$, it becomes:

$$\begin{aligned}
(aP+b)^q(cP+d) - (aP+b)(cP+d)^q &= \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} \beta(aP+b) - \alpha(cP+d) \qquad (E_m)\\
&= \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta a - \alpha c)P + (\beta b - \alpha d)\\
&= \lambda \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} P + \frac{\beta b - \alpha d}{\beta a - \alpha c}\\
&= \lambda \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} P - m^{-1}\cdot(\alpha:\beta),
\end{aligned}$$

where $P(X) - \infty$, if it appears, is by convention replaced by 1; indeed, when a/c is in $\mathbb{F}_q$, the expression $\beta a - \alpha c$ vanishes for a point $(\alpha : \beta) \in \mathbb{P}^1(\mathbb{F}_q)$ so that one of the factors of the product contains no term in $P(X)$.

The right-hand side is therefore, up to a multiplicative constant $\lambda \in \mathbb{F}_{q^2}^*$ a product of translates of the target $P(X)$ by elements of $\mathbb{F}_{q^2}$. We also see that if m belongs to $\mathrm{PGL}_2(\mathbb{F}_q)$, then it just acts as a permutation on the factors of the right-hand-side, and induces a relation which is identical to the original one. More generally, if m is in $\mathrm{PGL}_2(\mathbb{F}_{q^2})$ and h is in $\mathrm{PGL}_2(\mathbb{F}_q)$, then the equations associated to the matrices m and hm have the same right-hand sides, and there is no need to consider both. That is the reason why we have to consider the set of cosets $\mathcal{P}_q$. We expect the corresponding right-hand sides to behave independently with respect to linear dependencies (see heuristic below).

For the polynomial on the left-hand side of the equation, we need to reduce the degree of the systematic equation by using the special form of the defining polynomial: in $\mathbb{K}$ we have $X^q \equiv \frac{h_0(X)}{h_1(X)}$. Let us denote by a tilde the result of twisting elements of $\mathbb{F}_{q^2}$ by the Frobenius automorphism corresponding to q-th power computations. In particular, $\tilde{P}(X)$ is the polynomial $P(X)$ with all its coefficients raised to the power q. The left-hand side of the equation becomes

$$(\tilde{a}\tilde{P}(X^q) + \tilde{b})(cP(X) + d) - (aP(X) + b)(\tilde{c}\tilde{P}(X^q) + \tilde{d}),$$

and using the defining equation for the field $\mathbb{K}$, it is congruent to

$$\mathcal{L}_m = \left(\tilde{a}\tilde{P}\left(\frac{h_0(X)}{h_1(X)}\right) + \tilde{b}\right)(cP(X) + d) - (aP(X) + b)\left(\tilde{c}\tilde{P}\left(\frac{h_0(X)}{h_1(X)}\right) + \tilde{d}\right).$$

As a rational fraction, this expression $\mathcal{L}_m$ has a denominator of degree less than δD and a numerator of degree less than $(1 + \delta)D$. Since the denominator is a power of $h_1(X)$ and does not depend on m, from now on, when speaking about smooth elements, we assume that $h_1(X)$ is either indeed smooth, or has been added to the factor base elements. Hence, we are interested in the smoothness of the numerator of $\mathcal{L}_m$, and we say that $m \in \mathcal{P}_q$ yields a relation if this numerator of $\mathcal{L}_m$ is $\lceil D/2 \rceil$-smooth.

To any $m \in \mathcal{P}_q$, we associate a row vector $v(m)$ in the following way. Coordinates are indexed by $\lambda \in \mathbb{F}_{q^2}$, and the value associated to $\lambda \in \mathbb{F}_{q^2}$ is 1 or 0 depending on whether $P - \lambda$ appears in the right-hand side of the equation E_m. Equivalently, we may write

$$v(m)_{\lambda \in \mathbb{F}_{q^2}} = \begin{cases} 1 \text{ if } (\lambda : 1) = \kappa m^{-1} \cdot (\alpha : \beta) \text{ with } (\alpha : \beta) \in \mathbb{P}^1(\mathbb{F}_q) \text{ and } \kappa \in \mathbb{F}_{q^2}^*, \\ 0 \text{ otherwise.} \end{cases}$$

We associate to the polynomial P a matrix whose rows are the vectors $v(m)$ for the cosets $m \in \mathcal{P}_q$ which yield a relation. The validity of Proposition 1 crucially relies on the following heuristic.

Heuristic 3. *For any $P(X)$, the set of rows $v(m)$ for cosets $m \in \mathcal{P}_q$ that yield a relation form a matrix which has full rank.*

If this heuristic is verified, then a linear combination of the relations allows to rewrite $P(X)$ in terms of elements in the left-hand sides, that is polynomials of degree less than $\lceil D/2 \rceil$ as expected. Since there are $O(q^2)$ columns, the elimination process involves at most $O(q^2)$ rows, and since each row corresponds to an equation E_m, it involves at most D polynomials in the left-hand-side[1]. In total, the polynomial D is expressed by a linear combination of at most $O(q^2 D)$ polynomials of degree less than $\lceil D/2 \rceil$. This linear algebra step costs $O(q^5)$ using sparse matrix techniques. □

[1]This estimate of the number of irreducible factors is a pessimistic upper bound. In practice, one expects to have only $O(\log D)$ factors on average. Since the crude estimate does not change the overall complexity, we keep it that way to avoid adding another heuristic.

5 Supporting the heuristic argument in the proof

Heuristic 3 affects the validity of the algorithm described in the previous section. We propose two approaches to support this heuristic. Both allow to gain some confidence in the validity of the heuristic, but of course none affect the heuristic nature of this statement.

A first line of justification calls for another heuristic argument, similar to heuristics which underpin the analysis of many sub-exponential factorization or discrete logarithm algorithms. Heuristic 3 considers the rank of a matrix which depends on P. We denote this matrix by $H(P)$. Using notations introduced in Section 4, we remark that the rows of $H(P)$ are a subset of the set of row vectors $v(m)$ for $m \in \mathcal{P}_q$. We denote by $\mathcal{H}$ the larger matrix whose rows are all these vectors $v(m)$. The matrix $\mathcal{H}$ has $\#\mathcal{P}_q = q^3 + q$ rows and q^2 columns.

If the numerator of $\mathcal{L}_m$ behaves like a random polynomial of similar degree (which we might heuristically expect), then the probability that it is smooth is constant. Indeed, the smoothness bound is a constant times smaller than the total degree. More precisely, we want a polynomial of degree $(1+\delta)D$ to be $\lceil D/2 \rceil$-smooth, which occurs with probability close to $1/(2(1+\delta))!$. With δ being a constant, a constant proportion of the cosets m yield a relation, whence the matrix $H(P)$ is made of a constant fraction of the rows of the matrix $\mathcal{H}$. In other words, the matrix $H(P)$ has expected size $\theta(q^3) \times q^2$, and Heuristic 3 is related to the probability of $H(P)$ having maximal rank q^2. Unless some theoretical obstruction occurs, we expect the probability to have a matrix that is not full rank to be in $O(1/q)$. The matrix $\mathcal{H}$ is however peculiar, and does enjoy regularity properties which are worth noticing. For instance, we have the following proposition.

Proposition 4. *Let ℓ be a prime not dividing $q^3 - q$. Then the matrix $\mathcal{H}$ over $\mathbb{F}_\ell$ has rank q^2.*

Proof. If we extend $\mathcal{H}$ by one extra column corresponding to $\infty \in \mathbb{P}^1(\mathbb{F}_{q^2})$, thus forming a matrix $\mathcal{H}^+$, we have a bijection between rows of $\mathcal{H}^+$ and the different possible image sets of the projective line $\mathbb{P}^1(\mathbb{F}_q)$ within $\mathbb{P}^1(\mathbb{F}_{q^2})$, under injections of the form $(\alpha : \beta) \mapsto m^{-1} \cdot (\alpha : \beta)$. All these $q^3 + q$ image sets have size $q + 1$, and by symmetry all points of $\mathbb{P}^1(\mathbb{F}_{q^2})$ are reached equally often. Therefore, the sum of all rows of $\mathcal{H}^+$ is the vector whose coordinates all equal $\frac{1}{q^2}(q^3 + q)(q + 1) = q^2 + q$. The same holds for the matrix $\mathcal{H}$.

Let us now consider the sum of the rows in $\mathcal{H}^+$ whose first coordinate is 1 (as we have just shown, we have $q^2 + q$ such rows). Those correspond to image sets of $\mathbb{P}^1(\mathbb{F}_q)$ which contain one particular point, say $(0 : 1)$. The value of the sum for any other coordinate indexed by e.g. $Q \in \mathbb{P}^1(\mathbb{F}_{q^2})$ is the number of image sets $m^{-1} \cdot \mathbb{P}^1(\mathbb{F}_q)$ which contain both $(0 : 1)$ and Q, which we prove is equal to $q + 1$ as follows. Without loss of generality, we may assume $Q = \infty = (1 : 0)$. We need to count the number of possible cosets m. We need to count the homographies $m^{-1} \in \mathrm{PGL}_2(\mathbb{F}_{q^2})$, modulo $\mathrm{PGL}_2(\mathbb{F}_q)$-equivalence $m \equiv hm$, fixing $(0 : 1)$ and $(1 : 0)$. Letting $m^{-1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, we obtain $(b : d) = (0 : 1)$ and $(a : c) = (1 : 0)$, whence $b = c = 0$, and both $a, d \neq 0$. We may normalize to $d = 1$, and notice that multiplication of a by a scalar in $\mathbb{F}_q^*$ is absorbed in $\mathrm{PGL}_2(\mathbb{F}_q)$-equivalence. Therefore the number of suitable m is $\#\mathbb{F}_{q^2}^*/\mathbb{F}_q^* = q + 1$.

These two facts show that the row span of $\mathcal{H}^+$ (and, likewise, of $\mathcal{H}$) contains the vectors $(q^2+q, \ldots, q^2+q)$ and $(q^2 + q, q + 1, \ldots, q + 1)$. The vector $(q^3 - q, 0, \ldots, 0)$ is obtained as a linear combination of these two vectors, which suffices to prove that $\mathcal{H}^+$ and $\mathcal{H}$ have full rank, since the same reasoning holds for any coordinate. □

Proposition 4, while encouraging, is clearly not sufficient. We are, at the moment, unable to provide a proof of a more useful statement. On the experimental side, it is reasonably easy to sample arbitrary subsets of the rows of $\mathcal{H}$ and check for their rank. All trials we have made with random subsets of as few as q^2 rows have led to full rank matrices (experiments have been conducted up to $q = 2^5$).

A second line of justification is more direct and natural, as it is possible to implement the algorithm outlined in Section 4, and verify that it does provide the desired result. A Magma implementation validates this claim, and has been used to implement descent steps for an example field of degree 31 over $\mathbb{F}_{2^{10}}$. Of course this tiny example size uses no optimization, and is only intended to check the validity of Proposition 1.

6 Consequences for various ranges of parameters

Finding appropriate h_0 and h_1. One key fact about the algorithm is the existence of two polynomials h_0 and h_1 in $\mathbb{F}_q[X]$ such that $h_1(X)X^q - h_0(X)$ has an irreducible factor of degree k. A partial solution is due to Joux [Jou13b] who showed how to construct such polynomials when $k \in \{q-1, p, q+1\}$. No such deterministic construction is known in the general case, but experiments show that one can apparently choose h_0 and h_1 of degree at most 2. We performed an experiment for every odd prime power q in $[3, 1000]$ and every $k \leq q$ and found that we could select $a \in \mathbb{F}_{q^2}$ such that $X^q + X^2 + a$ has an irreducible factor of degree k. Finally, note that the result joins a commonly made heuristic in discrete logarithm algorithms: for fixed $f \in \mathbb{F}_{q^2}[X, Y]$ and random $g \in \mathbb{F}_{q^2}[X, Y]$, the polynomial $\mathrm{Res}_Y(f, g)$ behaves as a random polynomial of same degree with respect to the degrees of its irreducible factors.

Case where q is polynomial in the input size. The algorithm can be extended to the case where the base field is of the form $\mathbb{K} = \mathbb{F}_{q^k}$, with k prime, and q small, subject to the bound $q = O((\log q^k)^c)$. For this, the only option in order to fit within the range of applicability of Theorem 2 is to embed $\mathbb{K}$ in a larger field. Let q' be a power of q such that $q' > k$, and $\mathbb{K}'$ the corresponding superfield of $\mathbb{K}$. A discrete logarithm in $\mathbb{K}$ can be embedded in $\mathbb{K}'$ in polynomial time (isomorphisms of finite fields amount to polynomial factorizations). Then, by Theorem 2, the discrete logarithm in $\mathbb{K}'$ can be computed in time $q'^{O(\log k)}$. Let us rewrite this expression in term of the size of the input $Q = q^k$. The parameter q' is close to k, so that $q' = O(\log Q)$. Furthermore, we have $\log k = O(\log \log Q)$. In the end, we get a complexity for the discrete logarithm computation of $(\log Q)^{O(\log \log Q)}$, as in the case where q and k have similar sizes (but with a larger constant hidden in the $O()$).

Case where $q = L_{q^k}(\alpha)$. If the characteristic of the base field is not so small compared to the extension degree, the complexity of our algorithm does not keep its nice quasi-polynomial form. However, in almost the whole range of applicability of the Function Field Sieve algorithm, our algorithm is asymptotically better than FFS.

Let $Q = q^k$ the order of the base field, so that the input size is $\log Q = k \log q$. With our hypothesis, $\log q = (\log Q)^\alpha (\log \log Q)^{1-\alpha}$ and therefore $k = \log Q / \log q = O((\log Q / \log \log Q)^{1-\alpha})$.

The complexity of Theorem 2, takes the form $q^{O(\log k)} = L_Q(\alpha)^{O(\log \log Q)}$. This is smaller than $L_Q(\alpha')$ for any $\alpha' > \alpha$.

Hence, for any $\alpha < 1/3$, our algorithm is faster than the previously best known algorithm, namely FFS and its variant.

For $\alpha = 1/3$, the two rational side-variant of FFS provides an $L_Q(1/3)$ complexity, so that it remains better, just before being overtaken by the high-degree variant of NFS of [JLSV06]. More precisely, the limit between the two algorithm is around $q = \exp\left(O\left((\log(q^k)/\log\log(q^k))^{1/3}\right)\right)$.

We note that a generalized complexity expression which extends Theorem 2 to both cases discussed above is $O\left(\max(q, k)^{O(\log k)}\right)$.

7 An improvement based on additional heuristics

The Heuristic 3 tells that a rectangular matrix with $\theta(q)$ times more rows than columns has full rank. It seems reasonable to expect that only a constant times more rows than columns would be enough to get the full rank properties. Then, it means that we expect to have a lot of choices to select the best relations, in the sense that their left-hand sides split into irreducible factors of degrees as small as possible.

On average, we expect to be able to try $\theta(q)$ relations for each row of the matrix. So, assuming that the numerators of $\mathcal{L}_m$ behave like random polynomials of similar degrees, we have to evaluate the expected smoothness that we can hope for after trying $\theta(q)$ polynomials of degree $(1+\delta)D$ over $\mathbb{F}_{q^2}$. Set $u = \log q / \log \log q$, so that $u^u \approx q$. According to Theorem 1 in [PGF98] it is then possible to replace $\lceil D/2 \rceil$ in Proposition 1 by the value $O(D \log \log q / \log q)$.

Then, the discussion leading to Theorem 2 can be changed to take this faster descent into account. We keep the same estimate for the arity of each node in the tree, but the depth is now only in $\log k / \log\log q$. Since this depth ends up in the exponent, the resulting complexity in Theorem 2 is then

$$q^{O(\log k/\log\log q)}.$$

8 Conclusion

The algorithm presented in this paper achieves a significant improvement of the asymptotic complexity of discrete logarithm in finite fields, in almost the whole range of parameters where the Function Field Sieve was presently the most competitive algorithm. Compared to existing approaches, and in particular to the line of recent works [Jou13a, GGMZ13], the practical relevance of our algorithm is not clear, and will be explored by further work. Various optimization and tuning strategies can be used in this context.

- Because of the arity of the descent tree, the breadth eventually exceeds the number of polynomials below some degree bound. It makes no sense, therefore, to use the descent procedure beyond this point, as the recovery of discrete logarithms of all these polynomials is better achieved as a precomputation.
- The set of polynomials appearing in the right-hand side of equation E_m in Section 4 is $\{P-\lambda\}$, because in the factorization of $X^q - X$, we substitute X with $m \cdot P$ for homographies m. In fact, we may apply m to $(P_0 : P_1)$ for two polynomials $P_{0,1}$ (P being one of them). This leads to factors of the form $\lambda_1 P_0 - \lambda_0 P_1$ in the right-hand sides. This variant is expected to lead to a practical improvement, and also offers opportunities for obtaining better smoothness estimates.

We note that one of the key factors which hinders the practical efficiency of this algorithm is the $O(q^2 D)$ arity of the descent tree, compared to the $O(q)$ arity achieved by techniques based on Gröbner bases [Jou13a] at the expense of a $L(1/4+\epsilon)$ complexity.

The analysis of the algorithm presented here is heuristic, as discussed in Section 5. Some of the heuristics we stated, related to the properties of matrices $H(P)$ extracted from the matrix $\mathcal{H}$, seem accessible to more solid justification. It seems plausible to have the validity of algorithm rely on the sole heuristic of the validity of the smoothness estimates.

References

[Adl79] Leonard Adleman. A subexponential algorithm for the discrete logarithm problem with applications to cryptography. In *Foundations of Computer Science, 1979., 20th Annual Symposium on*, pages 55–60. IEEE, 1979.

[Adl94] Leonard Adleman. The function field sieve. In *Algorithmic number theory-ANTS I*, volume 877 of *Lecture Notes in Computer Science*, pages 108–121. Springer, 1994.

[Cop84] Don Coppersmith. Fast evaluation of logarithms in fields of characteristic two. *Information Theory, IEEE Transactions on*, 30(4):587–594, 1984.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976.

[GGMZ13] Faruk Göloglu, Robert Granger, Gary McGuire, and Jens Zumbrägel. On the function field sieve and the impact of higher splitting probabilities: Application to discrete logarithms in $\mathbb{F}_{2^{1971}}$. Cryptology ePrint Archive, Report 2013/074, 2013.

[Gor93] Daniel M Gordon. Discrete logarithms in GF(p) using the number field sieve. *SIAM Journal on Discrete Mathematics*, 6(1):124–138, 1993.

[JL06] Antoine Joux and Reynald Lercier. The function field sieve in the medium prime case. In *Advances in Cryptology-EUROCRYPT 2006*, volume 4005 of *Lecture Notes in Computer Science*, pages 254–270. Springer, 2006.

[JLSV06] Antoine Joux, Reynald Lercier, Nigel Smart, and Frederik Vercauteren. The number field sieve in the medium prime case. In *Advances in Cryptology-CRYPTO 2006*, pages 326–344. Springer, 2006.

[Jou13a] Antoine Joux. Faster Index Calculus for the Medium Prime Case Application to 1175-bit and 1425-bit Finite Fields. In *Advances in Cryptology–EUROCRYPT 2013*, pages 177–193. Springer, 2013.

[Jou13b] Antoine Joux. A new index calculus algorithm with complexity $L(1/4 + o(1))$ in very small characteristic. Cryptology ePrint Archive, Report 2013/095, 2013.

[PGF98] Daniel Panario, Xavier Gourdon, and Philippe Flajolet. An analytic approach to smooth polynomials over finite fields. In *Algorithmic number theory*, pages 226–236. Springer, 1998.