On the Function Field Sieve and the Impact of Higher Splitting Probabilities*

Application to Discrete Logarithms in $\mathbb{F}_{2^{1971}}$ and $\mathbb{F}_{2^{3164}}$

Faruk Göloğlu, Robert Granger, Gary McGuire, and Jens Zumbrägel

Complex & Adaptive Systems Laboratory and
School of Mathematical Sciences
University College Dublin, Ireland
{farukgologlu,robbiegranger}@gmail.com,
{gary.mcguire,jens.zumbragel}@ucd.ie

Abstract. In this paper we propose a binary field variant of the Joux-Lercier medium-sized Function Field Sieve, which results not only in complexities as low as $L_{q^n}(1/3, (4/9)^{1/3})$ for computing arbitrary logarithms, but also in an heuristic *polynomial time* algorithm for finding the discrete logarithms of degree one and two elements when the field has a subfield of an appropriate size. To illustrate the efficiency of the method, we have successfully solved the DLP in the finite fields with 2^{1971} and 2^{3164} elements, setting a record for binary fields.

Keywords: Discrete logarithm problem, function field sieve.

1 Introduction

When it comes to selecting appropriate parameters for public-key cryptosystems, one invariably observes a trade-off between security and efficiency. At a most basic level, for example, larger keys usually mean higher security, but worse performance.

A related rule of thumb which one does well to keep in mind, is that a specialised parameter which improves efficiency, typically (or potentially) weakens security. Examples abound of such specialisations and consequent attacks: discrete logarithms modulo Mersenne (or Crandall) primes and the Special Number Field Sieve [19]; Optimal Extension Fields [2] and Weil descent for elliptic curves [8]; high-compression algebraic tori [23] and specialised index calculus [10]; quasi-cyclic or dyadic McEliece variants [21] and Gröbner basis attacks [6], and more recently elliptic curves over binary fields [7], to name just a few. In practice therefore, one should be wary of any additional structure, which may potentially weaken a system.

* Research supported by the Claude Shannon Institute, Science Foundation Ireland Grant 06/MI/006. The fourth author was in addition supported by SFI Grant 08/IN.1/I1950.

In this paper we give a fairly extreme example of this principle in the case of binary (or in general small characteristic) fields which possess a small to medium-sized intermediate field. In 2006 Joux and Lercier designed a particularly efficient variation of the Function Field Sieve (FFS) algorithm for computing discrete logarithms [16], which at the time possessed the fastest asymptotic complexity of all known discrete logarithm algorithms for appropriately balanced q and n, namely $L_{q^n}(1/3, 3^{1/3}) \approx L_{q^n}(1/3, 1.442)$, where

$$L_{q^n}(a, c) = \exp\left((c + o(1))\,(\log q^n)^a (\log\log q^n)^{1-a}\right),$$

and q^n is the cardinality of the finite field.

In 2012, Joux proposed a more efficient method of obtaining relations, dubbed 'pinpointing', which applies to a specialisation of the function field setup of [16]. In this approach, each relation found via classical sieving can be amplified into many more [13], which is advantageous when sieving is the dominant phase, rather than the linear algebra (or individual logarithm phase). The overall complexity of this technique for solving the DLP can be as low as $L_{q^n}(1/3, (8/9)^{1/3}) \approx L_{q^n}(1/3, 0.961)$. To demonstrate the practicality of the approach, Joux solved the DLP in two cases: in a 1175-bit field and in a 1425-bit field, setting records for medium-sized base fields, in this case prime fields.

In this work we demonstrate that a basic assumption used in the analysis of virtually all fast index calculus algorithms can be very wrong indeed; in the case of binary fields possessing a subfield of an appropriate size, this leads to the dramatic conclusion that the logarithms of degree one elements over this subfield can be solved in *polynomial time*. As far as we are aware, no other algorithm for the collecting of relations and the linear algebra step has beaten the $L_{q^n}(1/3)$ barrier. Our fundamental observation is that the splitting probabilities in Joux-Lercier's variation of the FFS can be *cubic* in the reciprocal of the degree – rather than exponential. The reason for this is the richer structure of binary extension fields relative to prime fields, which lends weight to the argument that such fields should be avoided in practice. We also exploit our basic observation to efficiently compute the logarithms of degree two elements — which until now were the bottleneck of the individual logarithm descent phase — which for a range of binary fields results in the fastest $L_{q^n}(1/3)$ algorithm to date, namely $L_{q^n}(1/3, (4/9)^{1/3}) \approx L_{q^n}(1/3, 0.763)$, which is precisely the square root of the complexity of the ordinary FFS, for which $c = (32/9)^{1/3}$.

We emphasise that our relation generation method arises purely as a specialisation of [16], and is thus completely independent of [13]. However, at a high level, our relation generation method *may* be viewed as a form of one-sided pinpointing, but with two central differences to that of [13]. Firstly, we do not need to search for an initial splitting polynomial, since we have an explicit description of *all* such polynomials, i.e., no sieving need take place. Secondly, as members of this family of polynomials have arbitrarily high degree, the other 'random' side can be made to have very small degree, which thus splits with very high probability. These two differences result in our polynomial time relation generation.

The paper is organised as follows. In §2 we recall the Joux-Lercier variant of the FFS. In §3 we present our specialisation and our analysis of splitting

probabilities, while in §4 we present our new descent methods and analyse the complexity of the resulting algorithms. In §5 we present our implementation results and conclude in §6.

2 The Medium-sized Base Field Function Field Sieve

In this section we briefly recall the 2006 FFS variant of Joux and Lercier [16]. Let $\mathbb{F}_{q^n}$ be the finite field in which discrete logarithms are to be solved, where q is a prime power. In order to represent $\mathbb{F}_{q^n}$, choose two univariate polynomials $g_1, g_2 \in \mathbb{F}_q[X]$ of degrees d_1 and d_2 respectively. Then whenever $X - g_1(g_2(X))$ possesses a degree n irreducible factor $F(X)$ over $\mathbb{F}_q$, one can represent $\mathbb{F}_{q^n}$ in two related ways. In particular, let $x \in \mathbb{F}_{q^n}$ be a root of $F(X) = 0$, and let $y := g_2(x)$, so that by construction $x = g_1(y)$ as well. These relations give an explicit isomorphism between $\mathbb{F}_q(x)$ and $\mathbb{F}_q(y)$, both of which represent $\mathbb{F}_{q^n}$.

In the most basic version of the algorithm (which also leads to the best complexity) one chooses $d_1 \approx d_2 \approx \sqrt{n}$, and considers elements of $\mathbb{F}_{q^n}$ represented by:

$$xy + ay + bx + c\,, \quad \text{with} \quad a, b, c \in \mathbb{F}_q\,.$$

Substituting x by $g_1(y)$, and y by $g_2(x)$, we obtain the following equality in $\mathbb{F}_{q^n}$:

$$xg_2(x) + ag_2(x) + bx + c = yg_1(y) + ay + bg_1(y) + c\,. \tag{1}$$

The factor base consists simply of the degree one elements of $\mathbb{F}_q(x)$ and $\mathbb{F}_q(y)$; then for every triple (a, b, c) for which both sides of (1) split over $\mathbb{F}_q$ — i.e., when all of its roots are in $\mathbb{F}_q$ — in the respective factor bases, one obtains a relation. Determining such triples can naturally be made faster using sieving techniques. Once more than $2q$ such relations have been collected, one performs a linear algebra elimination to recover the individual logarithms. To compute arbitrary discrete logarithms, one uses a 'descent' method, as we detail in §4.

In order to assess the complexity of this algorithm, throughout the paper let $Q = q^n$, let $q = L_Q(1/3, \alpha)$, and let $L_Q(1/3, c_1)$ and $L_Q(1/3, c_2)$ denote the complexity of the sieving and linear algebra phases respectively. As shown in [16], heuristically one has

$$c_1 = \alpha + \frac{2}{3\sqrt{\alpha}} \quad \text{and} \quad c_2 = 2\alpha\,.$$

In order to generate sufficiently many relations, α must satisfy the condition:

$$2\alpha \geq \frac{2}{3\sqrt{\alpha}}\,.$$

For such α's, the complexity of the entire algorithm, including the descent phase, is minimised for $\alpha = 3^{-2/3}$, with resulting complexity $L_Q(1/3, 3^{1/3})$.

3 Specialisation to Binary Fields

We now present a specialisation of the construction of [16] as presented in the previous section, and detail some interesting consequences. From now on let $\mathbb{F}_q$ denote the finite field with 2^l elements.

All of our improvements and observations arise from a rather innocent-looking choice for g_2, namely $y = x^{2^k}$. Our primary motivation for this was to automatically eliminate half of the factor base, since any linear polynomial $(y + a)$ is equal to $(x + a^{2^{-k}})^{2^k}$, and so $\log(y + a) = 2^k \log(x + a^{2^{-k}})$. However, this selection has further serendipitous consequences, the central two being:

- Whenever $k \mid l$ and $l \geq 3k$, the probability of the l.h.s. of (1) splitting over $\mathbb{F}_q$ is approximately 2^{-3k}, instead of the expected $1/(2^k+1)!$. We show that for some asymptotic families of binary fields, this leads to a *polynomial time* algorithm to find the logarithms of all degree one elements of $\mathbb{F}_{q^n}$.
- As surprising as the above result is, for such families, the individual logarithm phase then has complexity $L_{q^n}(1/2)$. Hence one must ensure the complexity of the stages is balanced. Depending on the form of n, we show that the bottleneck of the descent changes from degree two to degree three special-$\mathfrak{q}$, since the x-side has the same form of the l.h.s. of (1), and thus enjoys the same higher splitting probability. This ensures that our claimed new $L_{q^n}(1/3)$ complexities are achieved across all the phases of the algorithm.

In the remainder of the paper we explain these advantages in more detail. In addition to the above two observations, for certain extensions which possess Galois-invariant factor bases, the use of non-prime base fields can induce extra automorphisms, which reduce its size further, see §5. Other practical speed ups arise from our choice $y = x^{2^k}$. The matrix-vector multiplications in Lanczos' algorithm consists of only cyclic rotations, i.e., shifts mod $q^n - 1$, and so no multiplications need to be performed. Furthermore, in the descent phase, one ordinarily needs to perform special-$\mathfrak{q}$ eliminations in both function fields. However, due to the simple relation between x and y, one is free to map from one side to the other in order to increase the probability of smoothness. One can also balance the degrees of both sides by utilising other auxiliary function fields arising from passing a power of 2 from the x-side to other side; this not only provides a practical speed up but is core to our new complexity results, see §4.

3.1 Higher Splitting Probabilities

Throughout this section, rather than use the field elements x, y as variables, we use X, Y to emphasise that the stated results are valid in the univariate polynomial ring over $\mathbb{F}_q$, which is implicitly either $\mathbb{F}_q[X]$ or $\mathbb{F}_q[Y]$, depending on which side of (1) is involved.

Assume $1 < k < l$. When $Y = X^{2^k}$ the l.h.s. of (1) becomes

$$X^{2^k+1} + aX^{2^k} + bX + c\,. \tag{2}$$

Assuming $c \neq ab$ and $b \neq a^{2^k}$, this polynomial may be transformed (up to a scalar factor) into the polynomial

$$f_B(\overline{X}) = \overline{X}^{2^k+1} + B\overline{X} + B\,, \quad \text{with} \quad B = \frac{(b + a^{2^k})^{2^k+1}}{(ab + c)^{2^k}}\,, \tag{3}$$

via

$$X = \frac{ab + c}{b + a^{2^k}}\,\overline{X} + a\,.$$

The polynomial f_B is related to $P_A(\overline{X}) = \overline{X}^{2^k+1} + \overline{X} + A$, which is well-studied in the literature, having arisen in several contexts including finite geometry, difference sets, as well as determining cross correlation between m-sequences; see references in [12] for further details.

We have the following theorem due to Bluher [3] (and refined in the binary case by Helleseth and Kholosha [12]), which counts the number of $B \in \mathbb{F}_q$ for which f_B splits over $\mathbb{F}_q$.

Theorem 1. *[12, Thm. 1] Let $d = \gcd(l, k)$. Then the number of $B \in \mathbb{F}_{2^l}^{\times}$ such that $f_B(\overline{X})$ has exactly $2^d + 1$ roots over $\mathbb{F}_{2^l}$ is*

$$\begin{cases} \dfrac{2^{l-d} - 1}{2^{2d} - 1} & \text{if } l/d \text{ odd,} \\ \dfrac{2^{l-d} - 2^d}{2^{2d} - 1} & \text{if } l/d \text{ even.} \end{cases}$$

Theorem 1 of [12] also states that f_B can have no more than $2^d + 1$ roots in $\mathbb{F}_q$, and so if $\gcd(l, k) < k$ then f_B can not split. Hence we must have $k \mid l$ for our application. Indeed we must also have $l \geq 3k$ in order for there to be at least one such B. Observe that under these two conditions, for B chosen uniformly at random from $\mathbb{F}_q$, the probability that f_B splits completely over $\mathbb{F}_q$ is approximately $1/2^{3k}$ – far higher than the splitting probability $1/(2^k + 1)!$ for a degree $2^k + 1$ polynomial chosen uniformly at random.

Furthermore, the set S_B of all such B can be computed explicitly, without needing to perform any factorisations or smoothness tests. Indeed, the proof of Prop. 5 in [12] gives an explicit parameterisation of all such B: for $u \in G = \mathbb{F}_{2^l} \setminus \mathbb{F}_{2^{2k}}$, we have

$$S_B = \mathrm{Im}\left(u \longrightarrow \frac{(u + u^{2^{2k}})^{2^k+1}}{(u + u^{2^k})^{2^{2k}+1}}\right).$$

By analysing the form of this map, one can avoid obtaining repeated images. However, even a naive enumeration of elements of G requires at most $\widetilde{O}(q)$ $\mathbb{F}_q$-operations, which is comparable to the complexity of relation generation, as we now show.

3.2 Relation Generation

By exploiting the above transformation of (2) to (3) and the list S_B of precomputed B's for which (3) splits, one can construct polynomials of the form (2) which always split completely over $\mathbb{F}_q$. Indeed, for any (a, b) for which $b \neq a^{2^k}$, and for each $B \in S_B$, we simply compute via (3) the corresponding unique $c \in \mathbb{F}_q$. This ensures that (2) splits and therefore requires no sieving whatsoever.

In order to obtain a relation, we also require that

$$Y g_1(Y) + b g_1(Y) + aY + c \tag{4}$$

splits over $\mathbb{F}_q$, which we assume occurs with probability $1/(d_1+1)!$ for randomly chosen g_1. Since $|L_B| \approx q/2^{3k}$, for each (a, b) we expect to obtain

$$\frac{q}{2^{3k}\,(d_1+1)!}$$

relations. Since we need q relations, we expect to require about $2^{3k}\,(d_1+1)!$ pairs (a, b) to obtain sufficiently many. For each pair (a, b) this costs $O(q/2^{3k})$ 1-smoothness tests, or $\tilde{O}(q/2^{3k})$ $\mathbb{F}_q$-operations. Hence the total cost is $\tilde{O}(q\,(d_1+1)!)$. Finally, in order for there to be sufficiently many relations, we must have

$$\frac{q^3}{2^{3k}\,(d_1+1)!} > q\,, \quad \text{or} \quad q^2 > 2^{3k}\,(d_1+1)!\,.$$

Since we insist that $l \geq 3k$, having $q > (d_1+1)!$ is sufficient. In §4 we consider the impact of this approach on the full DLP complexity in two cases when $q = L_{q^n}(1/3, \alpha)$ and $n \approx 2^k \cdot d_1$: firstly for $2^k \approx d_1$ and secondly for $2^k \gg d_1$. However, we now consider the relation generation complexity when the base field cardinality is polynomially related to the extension degree.

3.3 Polynomial Time Relation Generation

With a view to reducing the complexity of degree one relation generation to a minimum for some example fields, we choose k as large as possible such that $k \mid l$ and $l \geq 3k$, and set d_1 to be as small as possible, assuming a g_1 can be found with $X - g_1(X^{2^k})$ possessing a degree n irreducible factor. Experimentally it seems that $d_1 = 3$ (or possibly $d_1 = 4$) is sufficient to produce an irreducible of any degree $n \leq 2^k$, for q sufficiently large. Of course, n may be as large as $2^k \cdot d_1$ in this case.

Writing $l = k \cdot k'$ with $k' \geq 3$ a constant, and $n \approx 2^k \cdot d_1$ with d_1 constant, as $l \to \infty$, the logarithms of the degree one factor base elements of $\mathbb{F}_{q^n}$ can be computed in heuristic *polynomial time*. In particular, as $n \approx 2^k \cdot d_1 = 2^{l/k'} \cdot d_1$, we have

$$Q = q^n \approx 2^{l \cdot 2^{l/k'} \cdot d_1}\,.$$

As $l \to \infty$, we therefore have

$$\frac{\log Q}{\log\log Q} = O(2^{l/k'})\,.$$

The cost of relation generation is $\widetilde{O}(q\,(d_1+1)!) = \widetilde{O}(q) = \widetilde{O}(2^l) = \widetilde{O}(\log^{k'} Q)$, whereas the cost of sparse linear algebra, using Lanczos' algorithm [18] for instance, is the product of the row weight and the square of number of variables, namely

$$(2^{l/k'} + d_1)\,\widetilde{O}(q^2) = \widetilde{O}(\log^{2k'+1} Q)\,.$$

For the optimal choice $k' = 3$ the complexity is therefore $\widetilde{O}(\log^7 Q)$. We summarise this in the following:

Heuristic Result 1. *Let $q = 2^l$ with $l = k \cdot k'$ and $k' \geq 3$ a constant, let $d_1 \geq 3$ be constant, and assume $n \approx 2^k \cdot d_1$. Assuming that $Yg_1(Y)+aY+bg_1(Y)+c$ splits over $\mathbb{F}_q$ with probability $1/(d_1+1)!$ over all triples $(a,b,c) \in (\mathbb{F}_q)^3$, the logarithms of all degree one elements of $\mathbb{F}_{q^n}$ can be computed in time $\widetilde{O}(\log^{2k'+1} Q)$.*

Note that the set of degree one elements is always defined relative to a particular representation of $\mathbb{F}_{q^n}$. As it is easy to switch between any two representations of a finite field [20], one can always map to our $\mathbb{F}_q(x)$ first. Note also that the statement of Heuristic Result 1 implicitly assumes that the factor base contains a generator of $\mathbb{F}_{q^n}^{\times}$. A result of Chung proves that for all prime powers s and all $r \geq 1$ such that $s > (r-1)^2$, if $\mathbb{F}_{s^r} = \mathbb{F}_s(x)$ then $\{x + a \mid a \in \mathbb{F}_s\}$ generates $\mathbb{F}_{s^r}^{\times}$ [4, Thm. 8]. In our context we therefore need $q^{k'} > (n-1)^2 \approx q^2 \cdot d_1^2$ in order for our DLP algorithm to work, which is satisfied for our q and small d_1. However, the issue of whether there exists a generator in the stated factor base remains an open problem in general, see for instance [26].

3.4 An Extreme Case: $n = 2^k \pm 1$

If $n = 2^k \pm 1$ then the degree one relation generation becomes extremely fast. In particular, if $g_1(X) = \gamma X^{\mp 1}$ then as $g_2(X) = X^{2^k}$, we obtain the polynomials $X^{2^k \pm 1} + \gamma$. Furthermore, if $k \mid l$ then $X^{2^k \pm 1} + \gamma$ is irreducible whenever γ has no root of prime order $p \mid (2^k \pm 1)$. In both cases, (4) has degree two and splits with probability $1/2$.

Table 1 contains timing data for relation generation for a family of fields with $q = 2^{3k}$ and $n = 2^k - 1$, which incorporates the factor base reduction technique arising from quotienting out by the action of the k-th power of Frobenius, which has order $3n$, see §5. We used an AMD Opteron 6128 processor clocked at 2.0 GHz. Note that the time is quasi-cubic in the bitlengh, in accordance with the discussion preceeding Heuristic Result 1.

4 Individual Logarithms and Complexity Analysis

As unexpected as Heuristic Result 1 is, it does not by itself solve the DLP. Using a descent method à la [16, 5], computing individual logarithms unfortunately then has complexity $L_{q^n}(1/2)$. Hence one can not allow the extension degree n to grow as fast as Theorem 1 permits; it must be tempered relative to the base

Table 1. Relation generation times for $q = 2^{3k}$ and $n = 2^k - 1$

k	$\log_2(q^n)$	#vars	time
7	2667	5506	$2.3s$
8	6120	21932	$15.0s$
9	13797	87554	$122s$
10	30690	349858	$900s$

field size. With this in mind, we now consider the complexity of the descent, for q and n appropriately balanced so that the total complexity is $L_{q^n}(1/3)$.

For a generator $g \in \mathbb{F}_{q^n}^\times$ and a target element $h \in \langle g \rangle$, the descent proceeds by first finding an $i \in \mathbb{N}$ such that $z = h\, g^i$ is m-smooth for a suitable m, i.e., so that all of the irreducible factors of z have degrees $\leq m$. The goal of the descent is to eliminate every irreducible factor of z, by expressing each as a product of smaller degree irreducibles recursively, until only degree one elements remain, whose logarithms are known. We do so using the special-$\mathfrak{q}$ lattice approach from [16], as follows.

Let $p(x)$ be a degree d irreducible (considered as an element of $\mathbb{F}_q[X]$) which we wish to eliminate. Since $y = x^{2^k}$, we have

$$p(x)^{2^k} = \overline{p}(x^{2^k}) = \overline{p}(y)\,,$$

where the coefficients of $\overline{p}$ are those of p, powered by 2^k. Note that we also have

$$\overline{p}(y)^{2^{-k}} = p(x)\,,$$

and hence we can freely choose to eliminate p using either the x-side or the y-side of (1). For convenience we focus on the y-side. The corresponding lattice $L_{\overline{p}}$ is defined by:

$$L_{\overline{p}(Y)} = \{(w_0(Y), w_1(Y)) \in \mathbb{F}_q[Y]^2 : w_0(Y)\, g_1(Y) + w_1(Y) \equiv 0 \pmod{\overline{p}(Y)}\}\,.$$

A basis for this lattice is $(0, \overline{p}(Y)), (1, g_1(Y) \pmod{\overline{p}(Y)})$, which is clearly unbalanced. Using the extended Euclidean algorithm, we may construct a balanced basis $(u_0(Y), u_1(Y))$, $(v_0(Y), v_1(Y))$ for which the degrees are $\approx d/2$. Then for any $r(Y), s(Y) \in \mathbb{F}_q[Y]$ with $r(Y)$ monic we have

$$(w_0(Y), w_1(Y)) = \big(r(Y)u_0(Y) + s(Y)v_0(Y)\,,\; r(Y)u_1(Y) + s(Y)v_1(Y)\big) \in L_{\overline{p}(Y)}$$

and thus $\mathrm{RHS}(Y) \equiv 0 \pmod{\overline{p}(Y)}$, where

$$\mathrm{RHS}(Y) = w_0(Y)\, g_1(Y) + w_1(Y)\,.$$

When $\mathrm{RHS}(Y)/\overline{p}(Y)$ is $(d-1)$-smooth, we also check whether $\mathrm{LHS}(X)$ is also $(d-1)$-smooth, where

$$\mathrm{LHS}(X) = w_0(X^{2^k})\, X + w_1(X^{2^k})\,.$$

When both sides are $(d-1)$-smooth, we may replace $\overline{p}(Y)$ with a product of irreducibles of degree at most $d-1$, and then recurse.

Let $Q = q^n$. As in [16], we assume there is a parameter α such that:

$$n = \frac{1}{\alpha}\left(\frac{\log Q}{\log\log Q}\right)^{2/3}, \qquad q = \exp\left(\alpha \sqrt[3]{\log Q \cdot \log^2 \log Q}\right). \tag{5}$$

The three stages to consider are relation generation, linear algebra, and the descent, whose complexities we denote by $L_Q(1/3, c_1)$, $L_Q(1/3, c_2)$ and $L_Q(1/3, c_3)$, respectively. The total complexity is therefore $L_Q(1/3, c)$, where $c = \max\{c_1, c_2, c_3\}$. We next consider degree 2 elimination and then two special cases of field representation.

4.1 Degree 2 Elimination

We begin with degree 2 elimination as firstly it is the bottleneck in the descent, and secondly because one can exploit the higher splitting probability of the polynomials (2) as well. Let $\overline{p}(Y)$ be a degree 2 irreducible to be eliminated. A reduced basis $(u_0(Y), u_1(Y)), (v_0(Y), v_1(Y))$ for the lattice $L_{\overline{p}(Y)}$ can be found with degrees $(1,0), (0,1)$. Hence with r normalised to be 1 and $s \in \mathbb{F}_q$, we have

$$(w_0(Y), w_1(Y)) = \big(u_0(Y) + s\, v_0(Y), u_1(Y) + s\, v_1(Y)\big) \in L_{\overline{p}(Y)}$$

with degrees $(1,1)$. We have thus

$$w_0(Y)\, g_1(Y) + w_1(Y) \equiv 0 \pmod{\overline{p}(Y)},$$

and so the remaining factor has degree $d_1 - 1$. The corresponding polynomial $\mathrm{LHS}(X)$ is

$$w_0(X^{2^k})\, X + w_1(X^{2^k}), \tag{6}$$

which is of the form $X^{2^k+1} + aX^{2^k} + bX + c$, and as a consequence of Theorem 1, it splits over $\mathbb{F}_q$ with probability approximately 2^{-3k}. However, as with relation generation, we can also ensure that $\mathrm{LHS}(X)$ always splits, with the following technique. Writing the basis elements explicitly as $(Y + u_{00}, u_{10}), (v_{00}, Y + v_{10})$, and with $r = 1$ and $s \in \mathbb{F}_q$ the lattice elements are $(w_0(Y), w_1(Y)) = (Y + u_{00} + sv_{00}, sY + u_{10} + sv_{10})$. Thus combining (6) and (3), for each $B \in S_B$ we find the set of roots $s \in \mathbb{F}_q$ that satisfy the $\mathbb{F}_q[S]$ polynomial

$$B \cdot (v_{00}S^2 + (u_{00} + v_{10})S + u_{10})^{2^k} + (S^{2^k} + v_{00}S + u_{00})^{2^k+1} = 0,$$

by computing its GCD with $S^q + S$. This technique extracts all such s algebraically for any B, which ensures that $\mathrm{LHS}(X)$ automatically splits.

On average one expects there to be one such $s \in \mathbb{F}_q$ for each B. Then for each such s we check whether $\mathrm{RHS}(Y)/\overline{p}(Y)$ splits, which we assume occurs with probability $1/(d_1 - 1)!$. In general we therefore need sufficiently many B's in S_B for this to occur with good probability, i.e., that $q/2^{3k} > (d_1 - 1)!$.

4.2 Case 1: $n \approx 2^k \cdot d_1$ and $2^k \approx d_1$

In this section we will show the following:

Heuristic Result 2 (i). *Let $q = 2^l$, let $k \mid l$ and let n be such that (5) holds. Then for $n \approx 2^k \cdot d_1$ where $2^k \approx d_1$, the DLP can be solved with complexity $L_Q(1/3, (8/9)^{1/3}) \approx L_Q(1/3, 0.961)$.*

This is the simplest case we present; however for the sake of completeness and ease of exposition, we explicitly tailor the derivation presented in §3.2. By our relation generation method, the l.h.s. polynomial (2) always splits, whereas the probability of (4) being smooth is approximately $1/\sqrt{n}!$. Using the standard approximation $\log n! \approx n \log n$, the logarithm of the probability P of both sides being smooth is therefore:

$$\log P \approx -\sqrt{n} \log \sqrt{n} = -\frac{1}{2}\sqrt{n} \log n \, .$$

The size of the sieving space is $q^3/2^{3k}$, and since we require q relations we must have:

$$\frac{q^3 \, P}{2^{3k}} \geq q \, , \quad \text{or} \quad 2 \log q \geq \Big(\frac{3}{2} + \frac{\sqrt{n}}{2}\Big) \log n \approx \frac{\sqrt{n}}{2} \log n \, .$$

Ignoring low order terms, by (5) this is equivalent to

$$2\alpha \geq \frac{1}{3\sqrt{\alpha}} \, , \quad \text{or} \quad \alpha \geq 6^{-2/3} \, . \tag{7}$$

Given that we require q relations, the expected time to collect these relations is

$$\frac{q}{P} = L_Q\Big(1/3 \, , \, \alpha + \frac{1}{3\sqrt{\alpha}}\Big) \, ,$$

and hence $c_1 = \alpha + \frac{1}{3\sqrt{\alpha}}$. Since the linear algebra is quadratic in the size of the factor base, we also have $c_2 = 2\alpha$.

For the descent, as in [16], let the smoothness bound be $m = \mu\sqrt{n}$. Then the probability of finding such an expression is

$$1 \, / \, L_Q\Big(1/3 \, , \, \frac{1}{3\mu\sqrt{\alpha}}\Big) \, .$$

If the descent is to be no more costly than either the relation generation or the linear algebra, then we must have

$$\frac{1}{3\mu\sqrt{\alpha}} \; \leq \; \max\big\{\alpha + \frac{1}{3\sqrt{\alpha}} \, , \, 2\alpha\big\} \, . \tag{8}$$

We also need to ensure three further conditions are satisfied. Firstly, that the cost of all the special-$\mathfrak{q}$ eliminations is no more than $L_Q(1/3, \max\{c_1, c_2\})$. Secondly, that there are enough (r, s) pairs to ensure a relation is found. And thirdly, that

during the descent the degrees of the polynomials being tested for smoothness is really descending.

By the discussion in §4.1, in order to eliminate degree 2 elements we need $q \geq 2^{3k}\,(d_1-1)!$, or equivalently,

$$\alpha \geq \frac{1}{3\sqrt{\alpha}}\,, \quad \text{or} \quad \alpha \geq 3^{-2/3}\,.$$

Since for degree 3 special-$\mathfrak{q}$ LHS(X) will not have the form (2), we need to check that the smoothness probability does not impose an extra condition on α. For $\overline{p}(Y)$ a degree 3 irreducible to be eliminated, a reduced basis $(u_0(Y), u_1(Y))$, $(v_0(Y), v_1(Y))$ for the lattice $L_{\overline{p}(Y)}$ can be found with degrees $(1,1),(0,2)$. Hence with r now allowed to be monic of degree one and $s \in \mathbb{F}_q$, we have

$$(w_0(Y), w_1(Y)) = \big((Y+r_0)u_0(Y) + s\,v_0(Y)\,,\,(Y+r_0)u_1(Y) + s\,v_1(Y)\big) \in L_{\overline{p}(Y)},$$

with degrees $(2,2)$. As before, we have

$$w_0(Y)\,g_1(Y) + w_1(Y) \equiv 0 \pmod{\overline{p}(Y)}\,,$$

and the corresponding polynomial LHS(X) is

$$w_0(X^{2^k})\,X + w_1(X^{2^k})\,.$$

Once divided by $\overline{p}(Y)$, the degree of the Y-side is $d_1 - 1 \approx \sqrt{n}$ while the degree of the X-side is $2^{k+1}+1 \approx 2\sqrt{n}$. The logarithm of the probability that a degree n polynomial over $\mathbb{F}_q$ is m-smooth, for q and n tending to infinity but m fixed, can be estimated by $-(n/m)\log(n/m)$, as shown in [16]. Therefore the log of the probability P of both sides being 2-smooth is:

$$\log P \approx -\frac{\sqrt{n}}{2}\log\frac{\sqrt{n}}{2} - \frac{2\sqrt{n}}{2}\log\frac{2\sqrt{n}}{2} \approx -\frac{3}{2}\sqrt{n}\log\frac{\sqrt{n}}{2} \approx -\frac{3}{4}\sqrt{n}\log n\,,$$

and therefore $P = 1/L_Q(1/3, \frac{1}{2\sqrt{\alpha}})$. Since the (r,s) search space has size q^2 (which is also the complexity of the linear algebra), we require that

$$2\alpha \geq \frac{1}{2\sqrt{\alpha}} \quad \text{or} \quad \alpha \geq 16^{-1/3}\,.$$

Since $16^{-1/3} < 3^{-2/3}$, this imposes no additional constraint on α. Hence we can set $\alpha = 3^{-2/3}$, and one can check that in this case, $c_1 = c_2 = c_3 = 2\alpha$, giving complexity

$$L_Q(1/3, (8/9)^{1/3}) \approx L_Q(1/3, 0.961)\,,$$

which is precisely the complexity Joux obtained using either optimal one-sided, or advanced pinpointing [13]. Furthermore for this α, (8) implies that $\mu \geq 1/2$. For an upper bound, note that for special-$\mathfrak{q}$ of degree $\mu\sqrt{n}$, the degree of RHS(Y) is about $\sqrt{n}(1-\mu/2)$, while the degree of LHS(X) is about $\mu n/2$, so that $\mu < 2$ ensures that the descent is effective.

4.3 Case 2: $n \approx 2^k \cdot d_1$ and $2^k \gg d_1$

In this section we will show the following:

Heuristic Result 2 (ii). *Let $q = 2^l$, let $k \mid l$ and let n be such that (5) holds. Then for $n \approx 2^k \cdot d_1$ where $2^k \gg d_1$, the DLP can be solved with complexity between $L_Q(1/3, (4/9)^{1/3}) \approx L_Q(1/3, 0.763)$ and $L_Q(1/3, (1/2)^{1/3}) \approx L_Q(1/3, 0.794)$.*

Observe that interestingly, these two complexities are precisely the square-roots of the complexities of Coppersmith' algorithm [5], for which $c = (32/9)^{1/3}$ and $4^{1/3}$, the lower of the two being the complexity of the ordinary FFS [1, 14].

For n and q of the form (5), we claim that $c_1 = \alpha$, $c_2 = 2\alpha$, and that there are sufficiently many relations available. In particular, if we write $d_1 = n^\beta$ with $\beta < 1/2$ and $2^k = n^{1-\beta}$, then again by our relation generation method, the l.h.s. polynomial (2) always splits, and the log of the probability P of both sides being 1-smooth is:

$$\log P \approx -\beta n^\beta \log n.$$

By (5) we have

$$\begin{aligned} -\beta n^\beta \log n &\approx -\frac{2\beta}{3\alpha^\beta} \left(\frac{\log Q}{\log\log Q}\right)^{2\beta/3} (\log\log Q) \\ &= -\frac{2\beta}{3\alpha^\beta} (\log Q)^{2\beta/3} (\log\log Q)^{1-2\beta/3}. \end{aligned}$$

Hence the expected time of the relation generation is

$$\frac{q}{P} = L_Q(1/3, \alpha) \cdot L_Q\Big(2\beta/3, \frac{2\beta}{3\alpha^\beta}\Big).$$

For $\beta < 1/2$ the second term on the right is absorbed by the $o(1)$ term in the first term, and hence $c_1 = \alpha$ and $c_2 = 2\alpha$. The size of the sieving space is $q^3/2^{3k}$, and since we require q relations we must have:

$$\frac{q^3 P}{2^{3k}} \geq q\,, \quad \text{or} \quad L_Q(1/3, 2\alpha) \geq L_Q\Big(2\beta/3, \frac{2\beta}{3\alpha^\beta}\Big),$$

which holds for any $\alpha > 0$ when $\beta < 1/2$.

For the descent (as for Case 1) the cost of finding the first $\mu\sqrt{n}$-smooth relation is $L_Q(1/3, \frac{1}{3\mu\sqrt{\alpha}})$. And as before, for degree 2 special-$\mathfrak{q}$, the X-side has the same form and the condition on q arising from the search space being sufficiently large is always satisfied, since

$$q \geq 2^{3k} (d_1 - 1)! = n^{3(1-\beta)}\, L_Q\Big(2\beta/3, \frac{2\beta}{3\alpha^\beta}\Big),$$

which holds for any $\alpha > 0$ when $\beta < 1/2$.

Hence degree 3 special-$\mathfrak{q}$ are the bottleneck. As in the first case, with r allowed to be monic of degree one and $s \in \mathbb{F}_q$, the degree of RHS(Y) is $d_1 - 1$ while the

degree of LHS(X) is $2^{k+1}+1$. These degrees are clearly unbalanced. However, we can employ the following tactic to balance them.

Since $g_1(Y)^{2^k} + Y = 0$, we let $X' = g_1(Y)^{2^a}$ and thus $Y = X'^{2^{k-a}}$. We are free to choose any $1 < a < k$, as an elimination of a special-$\mathfrak{q}$ using Y and X' can be written in terms of Y and X by powering by a power of 2. With r allowed to be monic of degree one and $s \in \mathbb{F}_q$ we have $(w_0(Y), w_1(Y)) \in L_{\overline{p}(Y)}$ with degrees $(2,2)$, and our new expressions become

$$w_0(Y)\, g_1(Y)^{2^a} + w_1(Y) \equiv 0 \pmod{\overline{p}(Y)}\,.$$

The corresponding polynomial LHS(X') is

$$w_0(X'^{2^{k-a}})\, X' + w_1(X'^{2^{k-a}})\,.$$

Assuming the degrees are (approximately) the same, taking logs we have

$$k - a + 1 = \log_2(d_1) + a\,, \quad \text{or} \quad a = (k + 1 - \log_2(d_1))/2\,.$$

Since a must be an integer, rather than a real variable, we must choose the nearest integer to this value. In the best case, we can take a to be this exact value, and consequently both degrees are $\sqrt{2d_1}\, 2^{k/2} = \sqrt{2}\sqrt{n}$. Therefore the log of the probability P of both sides being 2-smooth is:

$$\log P \approx -\frac{\sqrt{2}}{2}\sqrt{n}\log\Big(\frac{\sqrt{2}}{2}\sqrt{n}\Big) - \frac{\sqrt{2}}{2}\sqrt{n}\log\Big(\frac{\sqrt{2}}{2}\sqrt{n}\Big) \approx -\frac{\sqrt{2}}{2}\sqrt{n}\log n\,,$$

and hence $P = L_Q(1/3, -\frac{\sqrt{2}}{3\sqrt{\alpha}})$. In order to have a sufficiently large search space we must therefore have

$$2\alpha \geq \frac{\sqrt{2}}{3\sqrt{\alpha}}\,, \quad \text{or} \quad \alpha \geq 18^{-1/3}\,.$$

For $\alpha = 18^{-1/3}$ the descent initiation stipulates that $\mu \geq \alpha^{-3/2}/6 = 1/\sqrt{2}$, and any $\mu \in [1/\sqrt{2}, \sqrt{n})$ suffices. We therefore have a total complexity of

$$L_Q(1/3, 2\alpha) = L_Q(1/3, (4/9)^{1/3}) \approx L_Q(1/3, 0.763)\,.$$

On the other hand when we need to round a to the nearest integer, the degrees can become unbalanced so that the degree of one side is up to double the degree of the other. In this case a simple calculation shows that the optimal α is $16^{-1/3}$, giving a complexity of

$$L_Q(1/3, 2\alpha) = L_Q(1/3, (1/2)^{1/3}) \approx L_Q(1/3, 0.794)\,.$$

Naturally, for a ratio of degrees in $(1/2, 2)$, we get c-values in between. This situation is redolent of Coppersmith's algorithm [5], in which precisely the same issue arises when forcing a real variable to take integer arguments only.

Note that this degree balancing technique also works for special-$\mathfrak{q}$ of any degree, making the descent far more rapid than for Case 1.

Remark 1. Observe that the best-case complexity with $c = (4/9)^{1/3}$ is precisely the complexity of the oracle-assisted Static Diffie-Hellman Problem in finite fields of small characteristic [17, §3]. Our result may therefore seem unsurprising, since the complexity of computing the logarithms of the factor base elements is never more than the complexity of the descent, and is thus effectively free. However, this reasoning overlooks the fact that we are working with a medium-sized base field, as opposed to the traditional FFS setting with a very small base field. In contrast to the result in [17, §3], our complexities depend crucially on our degree two elimination method, in addition to the fast computation of degree one logarithms.

5 Application to the DLP in $\mathbb{F}_{2^{1971}}$ and $\mathbb{F}_{2^{3164}}$

In this section we provide details of our implementation for discrete logarithm computations in the finite fields with 2^{1971} (as announced in [9]) and 2^{3164} elements, respectively.

5.1 Discrete Logarithms in $\mathbb{F}_{2^{1971}}$

In order to represent the finite field with 2^{1971} elements we first defined $\mathbb{F}_q = \mathbb{F}_{2^{27}}$ by $\mathbb{F}_2[T]/(T^{27}+T^5+T^2+T+1)$. Denoting by t a root of this irreducible in $\mathbb{F}_{2^{27}}$ we defined $\mathbb{F}_{2^{1971}} = \mathbb{F}_{q^{73}}$ by $\mathbb{F}_q[X]/(X^{73}+t)$. For x a root of $X^{73}+t$ in $\mathbb{F}_{q^{73}}$, we defined y by $y := x^8$, and we therefore also have $x = t/y^9$.

Since we use a Kummer extension, the elements of the factor base are related via the generator of the Galois group of $\mathbb{F}_{q^{73}}/\mathbb{F}_q$ [16, 13], and one can therefore quotient out by the action of this automorphism to reduce the number of variables from 2^{27} to $\approx 2^{27}/73$. As stated in §3, we can take this idea even further. In fact, $x^{2^9} = c\,x$ for $c = t^7 \in \mathbb{F}_q$, so the map $\sigma : a \to a^{2^9}$ is an additional automorphism which preserves the set of degree one factor base elements. The map σ^3 equals the Frobenius $a \to a^q$ (of order 73) and hence σ generates a group G of order 219. Considering the orbits of G acting on the factor base elements, we find 612 864 orbits of full size 219, seven of size 73, and one of size 1, resulting in $N = 612\,872$ orbits, which gives the number of factor base variables.

Since the degrees of the polynomials relating x and y are nearly balanced, the complexity of our relation generation falls into Case 1 in §4.2, which matches Joux's optimal one-sided, or advanced pinpointing for Kummer extensions. However, for Kummer extensions for which the degrees are balanced — as opposed to being very skewed as in §3.4 where $2^k \gg d_1$ — the advanced pinpointing is faster in practice, and so we used it for relation generation. We computed approximately $10N$ relations in about 14 core-hours computation time. For simplicity, we keep only those relations with distinct factors; this ensures that each entry of the relation matrix is a power of two, and hence all element multiplications in the matrix-vector products consist of cyclic rotations modulo $2^{1971}-1$.

After relation generation, we performed structured Gaussian elimination (SGE) (in a version based on [15]) to reduce the number of variables and thus to

decrease the cost for the subsequent linear algebra step. During our experiments we made the observation that additional equations are indeed useful for reducing the number of variables. However, the benefit of SGE is unclear as the row weight is being increased. We therefore stopped the SGE at this point, which resulted in a $528\,812 \times 527\,766$ matrix of constant row weight 19. The running time here was about 10 minutes on a single core.

We obtained the following partial factorisation of $2^{1971} - 1$:

$$7 \cdot 73^2 \cdot 439 \cdot 3943 \cdot 262657 \cdot 2298041 \cdot 10178663167 \cdot 27265714183 \cdot 9361973132609$$
$$\cdot 1406791071629857 \cdot 5271393791658529 \cdot 671165898617413417 \cdot 2762194134676763431$$
$$\cdot 4815314615204347717321 \cdot 421859275529837631474313737 19$$
$$\cdot 2206836284671480716039792791233921644 1$$
$$\cdot 781335393705318202869110024684359759405179097 \cdot C_{338} ,$$

where C_{338} is a 338-digit composite. We took as our modulus for the linear algebra step the product of C_{338} and the six largest prime factors of the cofactor, which has 507 digits. We applied a parallel version of Lanczos' algorithm (see [18]) using OpenMP on an SGI Altix ICE 8200EX cluster using Intel (Westmere) Xeon E5650 hex-core processors and GNU Multi-Precision library [11], taking 2220 core-hours in total.

For the DLP we took as (a presumed) generator $g = x + 1 \in \mathbb{F}_{2^{1971}}^{\times}$ and the target element was set as usual to be

$$x_\pi = \sum_{i=0}^{72} \tau(\lfloor \pi \cdot q^{i+1} \rfloor \bmod q)\, x^i ,$$

where τ takes the binary representation of an integer and maps to $\mathbb{F}_q$ via $2^i \mapsto t^i$. We first solved the target logarithm in the subgroups of order the first 11 terms in the factorisation using either linear search or Pollard's rho [22].

The descent proceeded by first finding an $i \in \mathbb{N}$ such that

$$x_\pi\, g^i = z_1/z_2 ,$$

where both z_1 and z_2 were 7-smooth. We implemented the descent in such a way that at the early phase of the algorithm the expected subsequent costs are as small as possible. This means that we try to find factorisations which consist of as many small degree factors as possible. We used about 40 core-hours to find an exponent i with favourable factorisation patterns and found $i = 47\,147\,576$ to be a good choice. We then spent about 3 hours to perform the descent down to degree 3. As stated in §3 and §4, at each stage during the descent, we can eliminate a given special-$\mathfrak{q}$ on either the x-side or on the y-side, one of which may be much faster. Computing the elimination probabilities we found that eliminating on the y-side is always faster. Indeed, for degree 2 special-$\mathfrak{q}$ we *must* perform this on the y-side, as it is not possible to do so on the x-side, due to the factorisation patterns of (2).

At this point we were left with 103 special-$\mathfrak{q}$ of degree 3, as opposed to the ≈ 500 expected with a random 7-smooth split of $x_\pi\, g^i$. The expected cost of

eliminating each of these is $2^{25.1}$ 2-smoothness tests. These special-$\mathfrak{q}$ elements were resolved on the same SGI Altix ICE 8200EX cluster in about 850 core-hours, using Shoup's Number Theory Library [24], resulting in 1140 special-$\mathfrak{q}$ elements of degree 2. Using the technique of §4.1, we reduced the cost of the elimination of each of these by a factor of $2^9 = 2^{3k}$, and all their logarithms were computed in 5 core-hours, completing the descent.

Thus the running time for solving an instance of the discrete logarithm problem completely in the finite field $\mathbb{F}_{2^{1971}}$ sums to $14+2220+898 = 3132$ core-hours in total. Finally, we found that $\log_g(x_\pi)$ equals

11992984215354106866091146371988855845186852755447163352368959007609021987957457840081811487759339446560383051978254174236023653588993736220077111736167826942310116340313535552228080411390321527355590590108228224824002192878782073040285652805730965886882790044168351003440859619124270006012898643375211000221438028988754606112522458797119787275080584651962314043764573936293823541736161168108256277804596578927095611589241735794006747396843460629926829429195737822645118262078374534950250296013992745319648974006524479548958327920827882768332440907342446643941097670216203953951337767311548343 .

5.2 Discrete Logarithms in $\mathbb{F}_{2^{3164}}$

For this case we defined $\mathbb{F}_q = \mathbb{F}_{2^{28}} = \mathbb{F}_2[T]/(T^{28}+T+1)$. We denote by t a root of this irreducible in $\mathbb{F}_{2^{28}}$. Furthermore, let $\mathbb{F}_{q^{113}} = \mathbb{F}_q[X]/(X^{113}+t)$ and denote by x a root of $X^{113}+t$ in $\mathbb{F}_{2^{3164}}$. We defined y by $y = x^{16}$, and we therefore also have $x = t/y^7$.

As in the previous section we use the Kummer extension idea of [16, 13] to reduce the size of the factor base. Again we can use a larger group than just the Galois group of $\mathbb{F}_{q^{113}}/\mathbb{F}_q$, since $x^{2^{14}} = c\,x$ for $c = t^9+t^8+t^5+t^4 \in \mathbb{F}_q$ and thus the map $\sigma : a \to a^{2^{14}}$ is an additional factor base preserving automorphism. The map σ^2 equals the Frobenius $a \to a^q$ and hence σ generates a group G of order 226. Considering the orbits of G acting on the factor base elements, we find $N = 1\,187\,841$ orbits in total, which gives the number of factor base variables.

For relation generation, since $16 > 7$ the degrees are unbalanced and hence more favourable toward the use of our relation generation method as given in §3.2. It produces one relation in just under a second, so that more than N relations can be found in about 350 core-hours. However, thanks to our choice of g_2, Joux's pinpointing methods *also* benefit from the higher splitting probability as explained by Theorem 1, and so for this Kummer extension, it is still preferable to use Joux's advanced pinpointing method, which generates about $10N$ relations in approximately 2 hours on a single-core.

With the structured Gaussian elimination step in mind we computed approximately $10N$ relations and performed SGE on this matrix to reduce the number of variables, where we stopped again at the point when the row weight is being

increased. The result was a $1\,066\,010 \times 1\,064\,991$ matrix of constant row weight 25, which constitutes a reduction of 10.3% in the number of variables.

The full factorisation of $2^{3164}-1$ (obtained from the Cunningham tables [25]) is:

$$3 \cdot 5 \cdot 29 \cdot 43 \cdot 113^2 \cdot 127 \cdot 227 \cdot 1583 \cdot 3391 \cdot 6329 \cdot 23279 \cdot 48817 \cdot 58309 \cdot 65993 \cdot 85429$$
$$\cdot 1868569 \cdot 2362153 \cdot 116163097 \cdot 636190001 \cdot 7920714887 \cdot 54112378027$$
$$\cdot 1066818132868207 \cdot 94395483835364237 \cdot 362648335437701461 \cdot 491003369344660409$$
$$\cdot 15079116213901326178369 \cdot 10384593717069655112945804582584321$$
$$\cdot 16210807687504089730597044158159945072569569899134297641 53$$
$$\cdot 25492807273453795564805967522921896342698297652509936704025490424226 49$$
$$\cdot 47852903674919527709794449504727427687484814404052312692462789051543 17$$
$$\cdot 94732691570793956856759198414911779734119524415635396799864941098330965560269355785101434237$$
$$\cdot 30893732435679706159469738259014519623666572271820219584074344744581789677891394468799700226702382646061113258175500479 9$$
$$\cdot 33248138195822034659908271092377125566098001373614163921550203376275101358208879881599077605921097512410793579836318474132090869696712 1 \cdot P_{190}\,,$$

where P_{190} is a 190-digit prime.

We then ran a parallel version of the Lanczos' algorithm on several nodes of the SGI Altix ICE 8200EX cluster, using MPI and OpenMP parallelisation techniques on 144 cores and again the GNU Multi-Precision library [11], taking 85488 core-hours in total. Note that since the nodes we used for the computation were not very "well-connected," the total running time would have been reduced to around 30000 core-hours if we had run our algorithm on 12 cores.

For the DLP we took as our (proven) generator $g = x + t + 1 \in \mathbb{F}_{2^{3164}}^{\times}$ and a target element set as usual to be $x_\pi = \sum_{i=0}^{113} \tau(\lfloor \pi \cdot q^{i+1} \rfloor \bmod q)\, x^i$.

As before the descent proceeded by first finding an $i \in \mathbb{N}$ such that $x_\pi\, g^i = z_1/z_2$, where both z_1 and z_2 were here 16-smooth. At each stage, we choose to sieve for the special-$\mathfrak{q}$ on the y-side.

In this case we put even more effort in analysing and optimising the descent in the earlier stages so that the expected subsequent costs will be minimised. In fact we associated a cost k_d to each factor of degree d arising in the factorisation of the l.h.s. and r.h.s. polynomials, which we estimated by considering the distribution of factorisation pattern.

We used about 70 core-hours to find the 16-smooth initial fraction z_1/z_2, then spent 210 core-hours for the descent down to degree 4, and used 340 core-hours for processing the degree 4 polynomials. At this point we had 71 polynomials of degree 3, which needed an expected number of $2^{34.1}$ 2-smoothness tests to be resolved. These special-$\mathfrak{q}$ elements have been processed by the same SGI Altix ICE 8200EX cluster in about 20972 core-hours, using Shoup's Number Theory Library [24], and resulted in 1239 special-$\mathfrak{q}$ elements of degree 2. Finally, using the technique in §4.1, these elements were eliminated in about 10 core-hours, completing the descent.

The running time for solving an instance of the discrete logarithm problem completely in the finite field $\mathbb{F}_{2^{3164}}$ sums to $350+85488+20972+210+340+10 = 107092$ core-hours (as already indicated, this figure would be reduced to around 52000 core-hours if Lanczos' algorithm was run on 12 cores). Finally, we found that $\log_g(x_\pi)$ equals

241095867208470377990120207726164220907051431328878753338580871702487845657
126883120634910367653233575538571774779776654573178495647701688094481773173
140524389502529386852264636049383546885561763318178634174789337030959840258
271899626361867369755406779988551274283201239012948389915300241739340043916
105822834002897204293036197694065337903255793451858773664350130030722091666
253172541070447948299781221019342860701064036544430331967753114646806335063
300203074234861067471668411998204544319176832353801982221924995804295426167
112306970795960798988644631100037393291558580412406942004555116148790387654
960490008429769544400790081908807239407134157724166048246419405503557398035
897999852593196954031439629768776850999887720870561741913055531864041654707
840433795403753200520891617150254756586728215941551355064840779765682398993
156390000024249110739956919350069293033670423070299581557636664993721204536
863038736714880164096355781178708892302786491643781 33 .

Observe that this computation also breaks the elliptic curve DLP for supersingular curves defined over $\mathbb{F}_{2^{791}}$, with embedding degree 4. However, since 791 is not prime, even before this break, such curves would not have been recommended, due to the potential applicability of Weil descent attacks [8].

6 Conclusion

We have presented and analysed new variants of the medium-sized base field FFS, for binary fields, which have complexities as low as $L_{q^n}(1/3, (4/9)^{1/3})$ for computing arbitrary logarithms. Furthermore, for fields possessing a subfield of an appropriate size, we have provided the first ever heuristic *polynomial time* algorithm for finding the discrete logarithms of degree one and two elements, which have both been verified experimentally. To illustrate the efficiency of the methods, we have successfully solved the DLP in the finite fields $\mathbb{F}_{2^{1971}}$ and $\mathbb{F}_{2^{3164}}$, setting a record for binary fields.

It would be interesting to know whether there are more general theorems on splitting behaviours for other polynomials arising during the descent, and also to what extent the known theorems apply to other characteristics.

Acknowledgements

The authors would like to extend their thanks to the Irish Centre for High-End Computing (ICHEC) — and Gilles Civario in particular — for their support throughout the course of our computations.

References

1. Adleman, L.M., Huang, M.D.A.: Function field sieve method for discrete logarithms over finite fields. Inform. and Comput. 151(1-2), 5–16 (1999)
2. Bailey, D.V., Paar, C., Sarkozy, G., Hofri, M.: Computation in optimal extension fields. In: Conference on The Mathematics of Public Key Cryptography, The Fields Institute for Research in the Mathematical Sciences. pp. 12–17 (2000)
3. Bluher, A.W.: On $x^{q+1}+ax+b$. Finite Fields and Their Applications 10(3), 285–305 (2004)
4. Chung, F.R.K.: Diameters and eigenvalues. J. Amer. Math. Soc. 2(2), 187–196 (1989)
5. Coppersmith, D.: Fast evaluation of logarithms in fields of characteristic two. IEEE Trans. Inform. Theory 30(4), 587–593 (1984)
6. Faugère, J.C., Otmani, A., Perret, L., Tillich, J.P.: Algebraic cryptanalysis of McEliece variants with compact keys. In: Gilbert, H. (ed.) EUROCRYPT 2010, LNCS, vol. 6110, pp. 279–298. Springer, Heidelberg (2010)
7. Faugère, J.C., Perret, L., Petit, C., Renault, G.: Improving the complexity of index calculus algorithms in elliptic curves over binary fields. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012, LNCS, vol. 7237, pp. 27–44. Springer, Heidelberg (2012)
8. Gaudry, P., Hess, F., Smart, N.P.: Constructive and destructive facets of Weil descent on elliptic curves. J. Cryptology 15(1), 19–46 (2002)
9. Göloğlu, F., Granger, R., McGuire, G., Zumbrägel, J.: Discrete Logarithms in $GF(2^{1971})$. NMBRTHRY list (19 Feb 2013)
10. Granger, R., Vercauteren, F.: On the discrete logarithm problem on algebraic tori. In: Shoup, V. (ed.) CRYPTO 2005, LNCS, vol. 3621, pp. 66–85. Springer, Heidelberg (2005)
11. Granlund, T., the GMP development team: GNU MP: The GNU Multiple Precision Arithmetic Library, 5.0.5 edn. (2012), `http://gmplib.org/`
12. Helleseth, T., Kholosha, A.: $x^{2^l+1} + x + a$ and related affine polynomials over GF(2^k). Cryptogr. Commun. 2(1), 85–109 (2010)
13. Joux, A.: Faster index calculus for the medium prime case application to 1175-bit and 1425-bit finite fields. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013, LNCS, vol. 7881, pp. 177–193. Springer, Heidelberg (2013)
14. Joux, A., Lercier, R.: The function field sieve is quite special. In: Fieker, C., Kohel, D.R. (eds.) Algorithmic number theory (Sydney, 2002), LNCS, vol. 2369, pp. 431–445. Springer, Heidelberg (2002)
15. Joux, A., Lercier, R.: Improvements to the general number field sieve for discrete logarithms in prime fields: a comparison with the gaussian integer method. Math. Comput. 72(242), 953–967 (2003)
16. Joux, A., Lercier, R.: The function field sieve in the medium prime case. In: Vaudenay, S. (ed.) EUROCRYPT 2006, LNCS, vol. 4004, pp. 254–270. Springer, Heidelberg (2006)
17. Joux, A., Lercier, R., Naccache, D., Thomé, E.: Oracle-assisted static diffie-hellman is easier than discrete logarithms. In: Parker, M.G. (ed.) Cryptography and Coding. LNCS, vol. 5921, pp. 351–367. Springer, Heidelberg (2009)
18. LaMacchia, B.A., Odlyzko, A.M.: Solving large sparse linear systems over finite fields. In: Menezes, A.J., Vanstone, S.A. (eds.) CRYPTO '90, LNCS, vol. 537, pp. 109–133. Springer, Heidelberg (1991)

19. Lenstra, A.K., Lenstra, Jr., H.W. (eds.): The development of the number field sieve, Lecture Notes in Mathematics, vol. 1554. Springer, Heidelberg (1993)
20. Lenstra, Jr., H.W.: Finding isomorphisms between finite fields. Math. Comp. 56(193), 329–347 (1991)
21. Misoczki, R., Barreto, P.S.: Compact McEliece keys from Goppa codes. In: Jacobson, Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) Selected Areas in Cryptography, LNCS, vol. 5867, pp. 376–392. Springer, Heidelberg (2009)
22. Pollard, J.M.: Monte carlo methods for index computation ($\mathrm{mod}\, p$). Math. Comp. 32(143), 918–924 (1978)
23. Rubin, K., Silverberg, A.: Torus-based cryptography. In: Boneh, D. (ed.) CRYPTO 2003, LNCS, vol. 2729, pp. 349–365. Springer, Heidelberg (2003)
24. Shoup, V.: NTL: A library for doing number theory, 5.5.2 edn. (2009), `http://www.shoup.net/ntl/`
25. Wagstaff, S., et al.: The Cunningham Project. `http://homes.cerias.purdue.edu/~ssw/cun/index.html`
26. Wan, D.: Generators and irreducible polynomials over finite fields. Math. Comp. 66(219), 1195–1212 (1997)