Elliptic Curve Discrete Logarithm Problem over Small Degree Extension Fields

Application to the Static Diffie-Hellman Problem on $E(\mathbb{F}_{q^5})$

Antoine Joux[1] and Vanessa Vitse[2]

[1] DGA and Université de Versailles Saint-Quentin, Laboratoire PRISM, 45 avenue des États-Unis, F-78035 Versailles cedex, France
antoine.joux@m4x.org

[2] Université de Versailles Saint-Quentin, Laboratoire PRISM, 45 avenue des États-Unis, F-78035 Versailles cedex, France
vanessa.vitse@prism.uvsq.fr

Abstract. In 2008 and 2009, Gaudry and Diem proposed an index calculus method for the resolution of the discrete logarithm on the group of points of an elliptic curve defined over a small degree extension field $\mathbb{F}_{q^n}$. In this paper, we study a variation of this index calculus method, improving the overall asymptotic complexity when $n = \Omega(\sqrt[3]{\log_2 q})$. In particular, we are able to successfully obtain relations on $E(\mathbb{F}_{q^5})$, whereas the more expensive computational complexity of Gaudry and Diem's initial algorithm makes it impractical in this case. An important ingredient of this result is a variation of Faugère's Gröbner basis algorithm F4, which significantly speeds up the relation computation. We show how this index calculus also applies to oracle-assisted resolutions of the static Diffie-Hellman problem on these elliptic curves.

Key words: elliptic curve, discrete logarithm problem (DLP), index calculus, Gröbner basis computation, summation polynomials, static Diffie-Hellman problem (SDHP)

1 Introduction

Given a finite group G and two elements $g, h \in G$, the *discrete logarithm problem* (DLP) consists in computing – when it exists – an integer x such that $h = g^x$. The difficulty of this problem is at the heart of many existing cryptosystems, such as the Diffie-Hellman key exchange protocol [12], the ElGamal encryption and signature scheme [14], DSA, or more recently in pairing-based cryptography. Historically, the DLP was first studied in the multiplicative group of finite fields. In such groups, now standard index calculus methods allow to solve the DLP with a subexponential complexity. Therefore, the key size necessary to achieve a given level of security is rather large. For this reason, in 1985, Miller [34] and Koblitz [30] suggested using for G the group of points of an elliptic curve, thus introducing algebraic curves to the cryptographic community.

Up to now, very few algorithms exist that solve the DLP in the group of points of an elliptic curve defined over a finite field (ECDLP). In most cases, only generic methods such as Baby-step Giant-step [42] or Pollard rho and kangaroo algorithms [37, 38] are available. Their complexity is exponential in the size of the largest prime factor of the group cardinality; more precisely, the running time is of the order of the square root of this largest prime factor [36]. However, for some specific curves more powerful attacks can be applied; they usually move the DLP to another, weaker group. The first approach is to lift the ECDLP to a characteristic zero field, either global (i.e. $\mathbb{Q}$) or local (i.e. p-adic numbers $\mathbb{Q}_p$): so far, this works only for subgroups of $E(\mathbb{F}_{p^n})$ of order p^i [39, 40, 44]. The second approach is to transfer via the Weil or Tate pairing the DLP on $E(\mathbb{F}_q)$ to $\mathbb{F}_{q^k}^*$: this includes the Menezes-Okamoto-Vanstone [33] and Frey-Rück [19] attacks for

elliptic curves with small embedding degree k. The last approach is to transfer via Weil descent the DLP on an elliptic curve defined over an extension field $\mathbb{F}_{q^n}$ to a second algebraic curve, defined over $\mathbb{F}_q$ but of greater genus g; this is efficient when the resulting genus g is small, which occurs only with specific curves [27].

In [41], Semaev proposed for the first time an index calculus method for the ECDLP, which unfortunately turned out to be impractical. However, combining Semaev's ideas and Weil restriction tools, Gaudry and Diem [11, 21] independently came up with an index calculus algorithm for elliptic curves defined over small degree extension fields, which has a better asymptotic complexity than generic algorithms. More precisely, the complexity of their algorithm over $E(\mathbb{F}_{q^n})$ for fixed n is in $\tilde{O}(q^{2-2/n})$, but with a hidden constant in n that grows over-exponentially, in $2^{O(n^2)}$. If one also allows n to go to infinity, then Diem shows that the complexity is subexponential as long as n is in $\Theta(\sqrt{\log_2(q)})$.

In this article, we investigate a variant of Gaudry and Diem's method and obtain the following result:

Theorem 1 *Let E be an elliptic curve defined over $\mathbb{F}_{q^n}$ and let G be a cyclic subgroup of its group of rational points. Then there exists an algorithm that solves the DLP in G and whose asymptotic complexity, under Assumptions 1 and 2, is*

$$\tilde{O}\left((n-1)!\left(2^{(n-1)(n-2)}e^n\, n^{-1/2}\right)^{\omega} q^2\right)$$

where ω is the effective complexity exponent of matrix multiplication.

Consequently, this new approach is asymptotically better than generic attacks like Pollard rho when $n \leq \frac{1}{2\omega}\log_2 q$, as q^n grows to infinity. Compared to Gaudry and Diem, it provides an asymptotic speed-up factor of $2^{(3-\omega)n^2}q^{-2/n}$, and hence is faster when $n \geq \left(\frac{2}{3-\omega}\log_2 q\right)^{1/3}$.

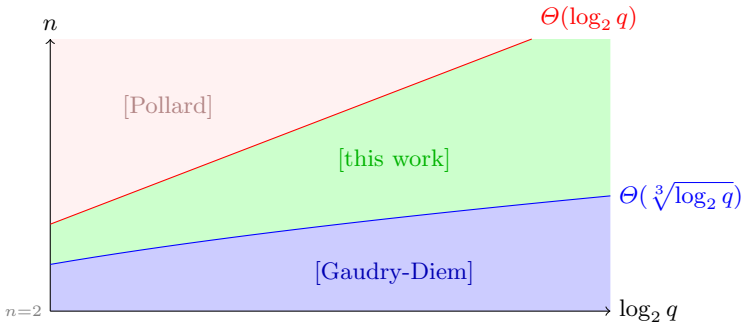

Fig. 1. Asymptotic comparison (for large values of q) of Pollard rho method, Gaudry and Diem's method and this paper for ECDLP over $\mathbb{F}_{q^n}, n \geq 2$.

The paper is organized as follows. First, we give a summary of Gaudry and Diem's index calculus; the main ideas are the use of the Weil restriction to obtain a convenient factor base and of Semaev's summation polynomials to test decompositions. More precisely, Gaudry and Diem check whether a given point can be decomposed as a sum of n points of this factor base, where n is the degree of the extension field. This amounts to solving a multivariate polynomial system of n equations in n variables of degree 2^{n-1}, arising from the $(n+1)$-th summation polynomial. Next, we introduce our variant: we check if a point decomposes as a sum of $n-1$ points instead of n. This reduces the likelihood of finding a relation, but greatly speeds up the decomposition process; as mentioned above, this trade-off is favorable when n is bigger than some multiple of $\sqrt[3]{\log_2 q}$. We then give a detailed analysis of the complexity of our variant, enabling us to prove

Theorem 1, and show that our trade-off is better than the one that could be provided by the hybrid approach of [4]. The following sections are devoted to several optimizations of the decomposition process. As already noted by Gaudry, the polynomial system that has to be solved is inherently symmetrical, so it pays off to reduce its total degree by writing down the equations in terms of the elementary symmetric functions before the resolution. A convenient way to do so is to use (partially) symmetrized summation polynomials instead of Semaev's; in section 3, we detail two different ways to directly compute these polynomials. The second optimization concerns the resolution of the symmetrized system. The fastest available method is to compute a Gröbner basis for an appropriate monomial order, using one of Faugère's algorithms [15, 16]. Since each relation search leads to a system with the same specific shape, we propose an ad hoc variant of F4 which takes advantage of this particularity to remove all reductions to zero. Finally, we present a variation of our algorithm which solves the oracle-assisted static Diffie-Hellman problem (SDHP, introduced in [6]) over $E(\mathbb{F}_{q^n})$. As in the case of finite fields presented in [28], solving the SDHP, after some oracle queries, is faster than solving the corresponding discrete logarithm problem. More precisely, we show that an attacker is able, after at most $q/2$ well-suited oracle queries, to compute an arbitrary SDHP instance reasonably quickly.

2 Index calculus algorithms for elliptic curves over extension fields

We begin by briefly recalling the principle of index calculus methods. We consider a finite abelian group G and two elements $h, g \in G$ such that $h = [x]g$ (in additive notation) where x is the secret to recover. For simplicity, we assume g has prime order ℓ. The basic outline consists of three main steps:

1. Choice of a factor base, i.e. a set $\mathcal{F} = \{g_1, \ldots, g_N\}$ of elements of G, generating the whole group G.
2. Relation search: for "random" integers $a_i, b_i \in \mathbb{Z}/\ell\mathbb{Z}$, try to decompose $[a_i]g + [b_i]h$ into the factor base[3], i.e. write
$$[a_i]g + [b_i]h = \sum_{j=1}^{N} [c_{ij}]g_j, \text{ where } c_{ij} \in \mathbb{Z}. \tag{1}$$
3. Linear algebra: once k relations of the form (1) have been found where k is large enough, construct the matrices $A = \begin{pmatrix} a_i & b_i \end{pmatrix}_{1 \le i \le k}$ and $M = (c_{ij})_{\substack{1 \le i \le k \\ 1 \le j \le N}}$, and find an element $v = \begin{pmatrix} v_1 & \ldots & v_k \end{pmatrix}$ in the left kernel of M such that $vA \neq \begin{pmatrix} 0 & 0 \end{pmatrix} \mod \ell$. Such an element v exists and can be computed with elementary linear algebra as soon as k is greater than N and the relations are linearly independent. The logarithm of h is then $x = -\left(\sum_i a_i v_i\right) / \left(\sum_i b_i v_i\right) \mod \ell$.

Other variants exist for steps 2 and 3, such as the precomputation-and-descent: all relations considered are of the form (1) with $b_i = 0$, and the linear algebra yields the logarithms in base g of all factor base elements; then only one relation involving h is needed to obtain its logarithm. This method is much more efficient when one has more than one discrete logarithm to compute in base g.

For example, if G is the multiplicative group of $\mathbb{F}_p$, p prime, we can take for $\mathcal{F}$ the set of equivalence classes of prime integers smaller than a fixed bound B. An element is then decomposable in this factor base if its representative in $[1, p-1]$ is B-smooth. There is obviously a compromise to be found: if B is large, then most elements are decomposable, but many relations are necessary and the matrices involved in the linear algebra step are comparatively large. On the other hand, if the factor base is small, the required number of relations is small and the linear algebra step is fast, but finding a relation is much less probable. In any case, the matrix M is usually very sparse and appropriate techniques can be used to compute its kernel quickly.

[3] For the complexity analysis, it is usually necessary to assume that the elements to decompose are chosen randomly in G. If the group generated by g is different from G, this can be achieved by considering elements $g_1, \ldots, g_t \in \mathcal{F}$ that together with g generate G. One then tries to decompose random combinations of the form $[a_i]g + [b_i]h + [c_{i,1}]g_1 + \ldots + [c_{i,t}]g_t$.

One major obstruction to index calculus when G is the group of rational points of an elliptic curve defined over $\mathbb{F}_q$, is that there exist no obvious factor bases. The second, related difficulty is that decomposing an element as in (1) is really not straightforward. In [41], Semaev proposes the first efficient way to find such decomposition, yet his approach could not work for lack of an adequate factor base.

2.1 The versions of Gaudry and Diem

In [11, 21], Gaudry and Diem propose to apply an index calculus method in the group of rational points of elliptic curves defined over small degree extension fields. To do so, they combine ideas from Semaev's index calculus proposal and Weil descent attack, to get a multivariate polynomial system that one can solve using Gröbner basis techniques. More precisely, if E is an elliptic curve defined over $\mathbb{F}_{q^n}$, Gaudry's choice of factor base is the set of points whose x-coordinate lies in the base field: $\{P \in E(\mathbb{F}_{q^n}) : P = (x_P, y_P),\ x_P \in \mathbb{F}_q,\ y_P \in \mathbb{F}_{q^n}\}$. Actually, since this set is invariant under negation, it is possible to consider only one half of it; if E is given in reduced Weierstrass form in characteristic different from 2 or 3, the factor base becomes:

$$\mathcal{F} = \{P \in E(\mathbb{F}_{q^n}) : P = (x_P, y_P),\ x_P \in \mathbb{F}_q,\ y_P \in S\}$$

where S is a subset of $\mathbb{F}_{q^n}$ such that $\mathbb{F}_{q^n} = S \cup (-S)$ and $S \cap (-S) = \{0\}$ (for example, assuming that -1 is not a square in $\mathbb{F}_{q^n}$, we can choose for S the set of quadratic residues together with 0). The same kind of twofold reduction can also be done for a general equation of E.

To compute the discrete logarithm of $Q \in \langle P \rangle$ with an index calculus algorithm, we first need to find *relations*, i.e. to decompose combinations of the form $R = [a]P + [b]Q$ where a, b are random integers, as sum of points in $\mathcal{F}$. Following Semaev's idea, Gaudry suggests to consider only relations of the form

$$R = \pm P_1 \pm P_2 \pm \ldots \pm P_n \tag{2}$$

where n is the degree of the extension field and $P_i \in \mathcal{F}$ $(1 \leq i \leq n)$. Getting such relations can be done by using a Weil restriction process. One considers $\mathbb{F}_{q^n}$ as $\mathbb{F}_q[t]/(f(t))$ where $f(t)$ is an irreducible polynomial of degree n over $\mathbb{F}_q$, in order to represent points $P = (x_P, y_P) \in E(\mathbb{F}_{q^n})$ by $2n$ coordinates: $x_P = x_{0,P} + x_{1,P}t + \ldots + x_{n-1,P}t^{n-1}$ and $y_P = y_{0,P} + y_{1,P}t + \ldots + y_{n-1,P}t^{n-1}$. Instead of writing down an equation with $(n+1)n$ unknowns from the decomposition (2), it is rather convenient to use Semaev's summation polynomials to get rid of the y_{P_i} variables. We recall here the definition and properties of such polynomials.

Proposition 2 *Let E be an elliptic curve defined over a field K. The m-th Semaev's summation polynomial is an irreducible symmetric polynomial $f_m \in K[X_1, \ldots, X_m]$, of degree 2^{m-2} in each variable, such that given $P_1 = (x_{P_1}, y_{P_1}),\ \ldots, P_m = (x_{P_m}, y_{P_m}) \in E(\overline{K}) \setminus \{\mathcal{O}_E\}$, we have*

$$f_m(x_{P_1}, \ldots, x_{P_m}) = 0 \Leftrightarrow \exists \epsilon_1, \ldots, \epsilon_m \in \{1, -1\}, \epsilon_1 P_1 + \ldots + \epsilon_m P_m = \mathcal{O}_E.$$

These summation polynomials can be effectively computed by induction, see [41] or section 3. At this point, we replace (2) by the equivalent equation

$$f_{n+1}(x_{P_1}, \ldots, x_{P_n}, x_R) = 0, \tag{3}$$

using the $(n+1)$-th summation polynomial $f_{n+1} \in \mathbb{F}_{q^n}[X_1, \ldots, X_{n+1}]$. The unknowns $x_{P_1}, \ldots, x_{P_n}$ actually lie in $\mathbb{F}_q$, so if we sort (3) according to powers of t, we obtain $\sum_{i=0}^{n-1} \varphi_i(x_{P_1}, \ldots, x_{P_n})\, t^i = 0$ where each φ_i is a symmetric polynomial over $\mathbb{F}_q$, of degree at most 2^{n-1} in each variable (and whose coefficients depend polynomially on x_R, a and b). This leads to a system of n symmetric polynomials, which, as advised

by Gaudry, can be written as a system of polynomials of total degree 2^{n-1} in terms of the elementary symmetric functions $e_1, \ldots, e_n$ of the variables $x_{P_1}, \ldots, x_{P_n}$:

$$\varphi_0(e_1, \ldots, e_n) = 0, \quad \varphi_1(e_1, \ldots, e_n) = 0, \quad \ldots \quad \varphi_{n-1}(e_1, \ldots, e_n) = 0. \tag{4}$$

Given the $(n+1)$-th summation polynomial, writing down such a system is almost immediate, but solving it is much more complicated. Actually, the resolution cost is hard to estimate precisely, but is at least polynomial in the degree of the corresponding zero-dimensional ideal, which according to [11] is generically equal to the Bézout bound $2^{n(n-1)}$. This over-exponential complexity causes the attack to be unfeasible for $n \geq 5$ on current personal computers.

In order to deduce a relation of the form (2) from a solution of the symmetrized equation (4), we first need to find the roots of the univariate polynomial $F(x) = x^n - e_1 x^{n-1} + \ldots + (-1)^n e_n$; this can easily be done using classical algorithms like Cantor-Zassenhaus's [20, chap. 14]. When F is split over $\mathbb{F}_q$, it suffices to construct the points of $\mathcal{F}$ whose x-coordinates are roots of F and then test the 2^n possible arrangements of signs until the decomposition of R is found. The cost of this desymmetrization and sign-finding step is negligible compared to the resolution of the polynomial systems involved. We remark that when F is split, it is *a priori* possible that some of its roots do not correspond to x-coordinates of points in $\mathcal{F}$: this is the case if the associated y-coordinates lie in a quadratic extension $\mathbb{F}_{(q^n)^2}$ but not in $\mathbb{F}_{q^n}$. However, these points belong to the subgroup $G' = \varphi(E'(\mathbb{F}_{q^n})) \subset E(\mathbb{F}_{q^{2n}})$, where $E'_{|\mathbb{F}_{q^n}}$ is the quadratic twist of E and φ is a $\mathbb{F}_{q^{2n}}$-isomorphism between $E'(\mathbb{F}_{q^{2n}})$ and $E(\mathbb{F}_{q^{2n}})$. The decomposition of R given by F is thus $R = \pm P_1 \pm \ldots \pm P_k \pm P_{k+1} \pm \ldots \pm P_n$ where $P_i \in \mathcal{F}$ if $i \leq k$ and $P_i \in G'$ otherwise. This can be rewritten as $R \mp P_1 \mp \ldots \mp P_k = \pm P_{k+1} \pm \ldots \pm P_n$, where the left-hand side is in $E(\mathbb{F}_{q^n})$ and the right-hand side in G'. Since the intersection of these two groups is reduced to $E(\mathbb{F}_{q^n})[2]$, this means that $\pm P_{k+1} \pm \ldots \pm P_n$ is a decomposition of a 2-torsion point in only $n-k$ points with x-coordinate in $\mathbb{F}_q$. While possible, the existence of such a decomposition for a given elliptic curve is highly unlikely, so that in practice we always get a decomposition in $\mathcal{F}$ when F is split. Note that the cost of sign-finding can be reduced by a factor of 2 by fixing the sign of P_1 to + and by considering the 2^{n-1} possible signs for the other points, stopping when the corresponding sum is either R or $-R$ (this is efficiently tested by looking at the abcissa of the sum $P_1 \pm P_2 \pm \ldots \pm P_n$).

Once we get enough equations like (2), we proceed to the linear algebra step. After collecting $k > \#\mathcal{F} \simeq q/2$ distinct (independent) relations of the form:

$$[a_i]P + [b_i]Q = \sum_{j=1}^{N} [c_{ij}]P_j, \quad \text{where } N = \#\mathcal{F}, \quad c_{ij} \in \{0; 1; -1\} \quad \text{and} \quad \sum_{j=1}^{N} |c_{ij}| = n,$$

we get a vector $A = \left(a_i \; b_i\right)_{1 \leq i \leq k}$ and a matrix $M = (c_{ij})$ which is very sparse since it has only n entries per row. It remains to find a element ${}^t v \in \ker({}^t M)$ such that $vA \neq 0 \bmod \ell$, which yields the discrete logarithm $x = -\left(\sum_i a_i v_i\right) / \left(\sum_i b_i v_i\right) \bmod \ell$ of Q in base P.

Complexity estimate of Gaudry and Diem's algorithm

As a preliminary step, we should check that $\mathcal{F}$ contains enough points. Heuristically, it is clear that there is approximately $q/2$ elements in the factor base; this statement can be made rigorous. The geometric object corresponding to $\mathcal{F}$ is $\mathcal{C} = \{P \in E(\mathbb{F}_{q^n}) \; : \; P = (x_P, y_P), \; x_P \in \mathbb{F}_q\} \cup \{\mathcal{O}_E\}$, which is the disjoint union (up to 2-torsion points) of $\mathcal{F}$ and $-\mathcal{F}$ together with the point at infinity. This is a projective curve contained in the Weil restriction $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$ of E, relatively to the extension $\mathbb{F}_{q^n}/\mathbb{F}_q$ (note that the GHS attack uses precisely the Jacobian of an irreducible component of $\mathcal{C}$ for the transfer of the DLP [22]). It turns out that it is quite easy to determine when $\mathcal{C}$ is irreducible and to bound its genus, so that we can estimate its number of

points. According to Diem, we ask that the curve E satisfies the following condition, which is a reformulation of Condition 2.7 of [11]:

Condition 3 *There exists a 2-torsion point $P = (x_P, y_P) \in E(\overline{\mathbb{F}_{q^n}})$ such that for all $i \in \{1, \ldots, n-1\}$, $\sigma^i(x_P) \neq x_P$ and $\sigma^i(x_P)$ is not the x-coordinate of a 2-torsion of point of E, where σ is the exponentiation by q.*

Technically, this is a condition on the Weierstrass equation of E (and not on E itself). Diem showed that it is always possible to find an equation for E such that this condition is satisfied, even if E is initially defined over a proper subfield of $\mathbb{F}_{q^n}$. Under this condition, the number of points in $\mathcal{C}$ is greater than $q+1-n2^{n+2}(\sqrt{q}+1)$ (see [11], proof of Proposition 4.11); as soon as $n \leq c \log_2 q$ where $c < 1/2$, this is greater than $q/2$ for q large enough. Thus we have the following proposition:

Proposition 4 *For any $\epsilon > 0$, there exists a constant $C > 0$ such that for any $q > C$ and $n < (1/2-\epsilon)\log_2 q$, the factor base $\mathcal{F}$ associated to an elliptic curve defined over $\mathbb{F}_{q^n}$ satisfying Condition 3 contains more than $q/4$ elements.*

Note that even when $n > \log_2(q)/2$, it is probable that the factor base $\mathcal{F}$ has more than $q/4$ points. In fact, the possible values for the cardinality of an irreducible curve defined over $\mathbb{F}_q$ are concentrated near $q+1$, so that it is rather unlikely that no equation of an elliptic curve $E_{|\mathbb{F}_{q^n}}$ would yield a factor base with enough elements.

The first main step of the algorithm consists of collecting around $\#\mathcal{F} \simeq q/2$ relations of the form (2) to build the matrix M. The $(n+1)$-th summation polynomial can be determined once for all using $\mathrm{Poly}(e^{(n+1)^2}\log_2 q)$ operations for a fixed elliptic curve [11]. The representation of this summation polynomial in terms of elementary symmetric functions can be done by using Gröbner elimination techniques for example, we refer to section 3 for improvements of this computation. The probability of finding a decomposition of a point $R \in E(\mathbb{F}_{q^n})$ is approximately

$$\frac{\#\mathcal{C}^n/\mathfrak{S}_n}{\#E(\mathbb{F}_{q^n})} \simeq \frac{q^n}{n!}\frac{1}{q^n} = \frac{1}{n!},$$

and the cost of checking if the point R is actually decomposable in the factor base, noted $c(n,q)$, is the cost of the resolution of a multivariate polynomial system of n equations defined over $\mathbb{F}_q$ with n variables of total degree 2^{n-1}. As we need at least $\#\mathcal{F}$ relations, the total complexity of the first step is about $n!\,c(n,q)\,q/2$.

The estimation of the cost $c(n,q)$ is not straightforward as it thoroughly depends on the algorithm used. Following Diem's analysis [11, section 4.5], the polynomial system considered is generically of dimension 0 since it has a finite number of solutions over $\overline{\mathbb{F}_q}$, and using resultant techniques we get an upper bound for $c(n,q)$:

$$c(n,q) \leq \mathrm{Poly}(n!\,2^{n(n-1)}\log_2 q).$$

The sparse linear algebra step can then be done in a time of $\tilde{O}(nq^2)$ with an adapted version of Lanczos or Wiedemann's algorithm [8, §20.3.3]. However, in order to improve the complexity of the algorithm, Gaudry suggests to rebalance the matrix-building cost against the linear algebra cost using "large primes" techniques adapted from [23] and [45] (see [21] for more details). By doing so, he needs to obtain approximately $q^{2-2/n}$ relations instead of q. The cost of the first main step becomes thus $n!\,c(n,q)\,q^{2-2/n}$. By contrast, the cost of the linear algebra step is reduced to $\tilde{O}(n\,q^{2-2/n})$, which is negligible compared to the previous step. As a result, the elliptic curve discrete logarithm problem over $\mathbb{F}_{q^n}$ for fixed n can be solved in an expected time of $\tilde{O}(q^{2-2/n})$, but the hidden constant grows extremely fast with n. A complete complexity estimate using Diem's bound is:

$$n!\,\mathrm{Poly}(n!\,2^{n(n-1)}\log_2 q)\,q^{2-2/n}.$$

2.2 Our version

The bad behaviour (over-exponential) in n occurring in the complexity of Gaudry and Diem's algorithm remains a serious drawback and makes their approach practical only for very small extension degrees, namely $n = 3$ or 4. Since the cost of the multivariate system resolution heavily depends on the degree of the summation polynomial, the complexity can be considerably improved by considering only decompositions of combinations $R = [a]P + [b]Q$ as sum of $(n-1)$ points in $\mathcal{F}$, instead of n points as in [11, 21]. Even though we are lowering the probability of getting such a decomposition when q grows, the gain is sufficient to make this approach realistic for $n = 5$.

We can solve the equation:

$$[a]P + [b]Q = \pm P_1 \pm \ldots \pm P_{n-1}, \tag{5}$$

where a and b are random integers and the P_i belong to $\mathcal{F}$, in the same way as explained in the previous section. The differences are that only the n-th summation polynomial is involved, and that the resulting system of n polynomials is in $(n-1)$ variables and of total degree only 2^{n-2}. Since this system is overdetermined, its resolution is greatly sped up, as compared to the previous case. The trade-off is that it is less probable to find a decomposition.

A toy example: we consider the curve

$$E : y^2 = x^3 + ax + b, \quad a = 60t^2 + 52t + 44,\ b = 74t^2 + 87t + 58$$

defined over $\mathbb{F}_{101^3} \simeq \mathbb{F}_{101}[t]/(t^3 + t + 1)$; it has prime order $\#E = 1029583$. Let $P = (84t^2 + 24t + 75, 92t^2 + 18t + 61)$ be a random generator of E and $Q = (50t^2 + 98t + 89, 2t^2 + 95t + 15)$.

We first try to decompose the random multiple $R = [47044]P + [956092]Q = (37t^2 + 84t + 85, 86t^2 + 3t + 15)$ as a sum of two points in $\mathcal{F}$. The third symmetrized Semaev's polynomial (see section 3) is

$$\tilde{f}_3(e_1, e_2, x_R) = (e_1^2 - 4e_2)x_R^2 - 2(e_1(e_2 + a) + 2b)x_R + (e_2 - a)^2 - 4be_1.$$

Replacing a, b and x_R by their respective values, we obtain the equation

$$(59t^2 + 29t + 100)e_1^2 + (27t^2 + 34t + 32)e_1e_2 + (31t^2 + 71t + 55)e_1 + e_2^2 + (48t^2 + 83t + 17)e_2 + 32t^2 + 16t + 81 = 0$$

whose Weil restriction yields the system

$$\begin{cases} 100e_1^2 + 32e_1e_2 + 55e_1 + e_2^2 + 17e_2 + 81 = 0 \\ 29e_1^2 + 34e_1e_2 + 71e_1 + 83e_2 + 16 = 0 \\ 59e_1^2 + 27e_1e_2 + 31e_1 + 48e_2 + 32 = 0 \end{cases}$$

This system has no solution, hence the point R (like most points of the curve) is not decomposable.

We then try to decompose the random multiple $R = [5620]P + [679359]Q = (16t^2 + 94t + 21, 80t^2 + 34t + 41)$. We obtain the equation

$$(61t^2 + 78t + 59)e_1^2 + (69t^2 + 14t + 59)e_1e_2 + (40t^2 + 20t + 57)e_1 + e_2^2 + (40t^2 + 89t + 80)e_2 + 12t^2 + 11t + 77 = 0$$

yielding the system

$$\begin{cases} 59e_1^2 + 59e_1e_2 + 57e_1 + e_2^2 + 80e_2 + 77 = 0 \\ 78e_1^2 + 14e_1e_2 + 20e_1 + 89e_2 + 11 = 0 \\ 61e_1^2 + 69e_1e_2 + 40e_1 + 40e_2 + 12 = 0 \end{cases}$$

This time the system has a unique solution $(e_1, e_2) = (69, 75)$, and the trinomial $X^2 - 69X + 75$ has two roots 6 and $63 \in \mathbb{F}_{101}$, corresponding to the points $P_1 = (6, 77t^2 + 93t + 35)$ and $P_2 = (63, t^2 + 66t + 2) \in \mathcal{F} \cup -\mathcal{F}$. We check that indeed $R = P_1 + P_2$. Since $\#(\mathcal{F} \cup -\mathcal{F}) = 108$, we need about 54 such decompositions to complete the relation search step. Once these relations are collected, we can deduce with sparse linear techniques that the discrete logarithm of Q in base P is $x = 715339$.

Complexity analysis

In order to estimate the complexity of our variant, we need to bound the probability that a random point is decomposable as a sum of $n-1$ points of $\mathcal{F}$ and to determine the cost of the resolution of the corresponding polynomial system.

Assumption 1 *The number of points of $E(\mathbb{F}_{q^n})$ that can be decomposed as a sum of $n-1$ elements of $\mathcal{F}$ is in $\Omega(q^{n-1}/(n-1)!)$.*

Let as before $\mathcal{C} = \{P \in E(\mathbb{F}_{q^n}) \ : \ P = (x_P, y_P),\ x_P \in \mathbb{F}_q\} \cup \{\mathcal{O}_E\}$. Since the cardinality of $\mathcal{C}^{n-1}/\mathfrak{S}_{n-1}$ is approximately $q^{n-1}/(n-1)!$, this assumption means that the preimage of most points of E under the map sending an unordered $(n-1)$-tuple of elements of $\mathcal{C}$ to their sum in $E(\mathbb{F}_{q^n})$ has a cardinality bounded by a constant independent of q and n.

Actually, this assumption is a corollary of the more precise following conjecture, as soon as the cardinality of $\mathcal{F}$ is in $\Omega(q)$ (which is always true provided that Condition 3 holds and that $n/(\log_2 q)$ stays bounded away from $1/2$, cf. Proposition 4 and the following discussion).

Conjecture. *Let $\mathcal{C}^{(n-1)} = \mathcal{C}^{n-1}/\mathfrak{S}_{n-1}$ be the $(n-1)$-th symmetric product of $\mathcal{C}$, and $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$ the Weil restriction of E relatively to the extension $\mathbb{F}_{q^n}/\mathbb{F}_q$. Let $\varsigma : (P_1, \ldots, P_{n-1}) \mapsto \sum_i P_i$ be the summation morphism $\mathcal{C}^{(n-1)} \to W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$, defined over $\mathbb{F}_q$. Then*

1. *the map ς is a degree one morphism from $\mathcal{C}^{(n-1)}$ to $\varsigma(\mathcal{C}^{(n-1)})$*
2. *the fiber $\varsigma^{-1}(R)$ of a point $R \in \varsigma(\mathcal{C}^{(n-1)})$ has positive dimension if and only if there exist $P_1, \ldots, P_{n-3} \in \mathcal{C}$ such that $R = \varsigma(P_1, \ldots, P_{n-3}, \mathcal{O}_E, \mathcal{O}_E)$.*

Part 1 of this conjecture can be verified formally for $n = 3$ using a computer algebra system, and has been satisfied for other extension degrees in all our examples. Note that $\mathcal{C}^{(n-1)}$ is a projective variety, so that in practice to check if this property holds, it is sufficient to find a point $R \in \varsigma(\mathcal{C}^{(n-1)})$ such that the fiber $\varsigma^{-1}(R)$ is a zero-dimensional variety of degree 1. Part 2 is less obvious to verify. It is clear that if $R = \varsigma(P_1, \ldots, P_{n-3}, \mathcal{O}_E, \mathcal{O}_E)$ (i.e. $R = \sum_{i=1}^{n-3} P_i$), then the corresponding fiber $\varsigma^{-1}(R)$ has dimension at least 1: it contains all the $(n-1)$-tuples of the form $(P_1, \ldots, P_{n-3}, Q, -Q)$. The converse is much more delicate, but has been verified experimentally (by exhaustive search) on small enough curves when $n = 5$. It is however reasonable to suppose that this conjecture holds, if not for all elliptic curves, at least for a large proportion of them (possibly up to a change of equation). In any case, with Assumption 1, the expected number of decomposition trials for the relation search is in $O((n-1)!\, q^2)$.

As mentioned above, the cost of trying to find one decomposition, noted $\tilde{c}(n, q)$, is reduced to the cost of the resolution of an overdetermined multivariate polynomial system of n equations with $(n-1)$ variables of total degree 2^{n-2}. Consequently, the complexity of the relation search step becomes $O((n-1)!\, \tilde{c}(n, q)\, q^2)$.

The value of $\tilde{c}(n, q)$ has to be compared to the cost $c(n-1, q)$: clearly we have $\tilde{c}(n, q) < c(n-1, q)$. Indeed, solving a system of n equations with $(n-1)$ variables of degree 2^{n-2} can be achieved by solving the system consisting of the first $(n-1)$ equations, and by checking the compatibility of the solutions with the last equation. With such an upper bound of $\tilde{c}(n, q)$, we obtain the complexity for the first collecting step of

$$O\left((n-1)!\, q^2 \mathrm{Poly}((n-1)!\, 2^{(n-1)(n-2)} \log_2 q)\right) \tag{6}$$

The linear algebra step has a complexity of $\tilde{O}(nq^2)$, which is negligible compared to the first step. Hence the total complexity of the algorithm is given by (6). We emphasize that because of the q^2 factor in the complexity, Pollard rho or other generic methods remain faster than our variant for $n \leq 4$, thus our approach is actually relevant only for $n \geq 5$. On the other hand, estimate (6) shows that there exists a constant c such that the variant is asymptotically faster than generic methods as soon as $5 \leq n \leq c \log_2 q$.

However, the resultant method yielding (6) is not optimal. A faster way of solving a zero-dimensional polynomial system over a finite field is to compute a lexicographic order Gröbner basis of the corresponding ideal. If this ideal is radical, up to a generic linear change of coordinates the resulting system of generators is of the form:

$$\{X_1 - g_1(X_n), \ldots, X_{n-1} - g_{n-1}(X_n), g_n(X_n)\},$$

where g_n is a univariate polynomial of degree equal to the degree of the ideal, and $g_1, \ldots, g_{n-1}$ are univariate polynomials of degree strictly smaller [3]. It is then easy to determine the set of solutions by finding the roots of g_n and then the corresponding values of the remaining variables. In the general case, the shape of the basis is not necessary linear in the remaining variables $X_1, \ldots, X_{n-1}$, but it is still simple to recover the solutions. Nevertheless, the computation of a lexicographic order Gröbner basis is usually difficult: this is not surprising since polynomial system solving is known to be a hard problem. For zero-dimensional ideals, an efficient strategy is to first compute a Gröbner basis for the graded reverse lexicographic order (*grevlex*) and then obtain the lex basis using a ordering change algorithm, such as FGLM [17]. We give some estimates of the complexity of the first stage later on; as for the second stage, its complexity is in $O(nD^3)$ field operation, where D is the degree of the ideal and n the number of variables.

In our case, the ideal is given by an overdetermined polynomial system. Generically, i.e. when $R \notin \varsigma(\mathcal{C}^{(n-1)})$, this ideal is the whole polynomial ring and the corresponding set of solutions is empty; its minimal Gröbner basis is $\{1\}$ for any order, including grevlex. Otherwise, i.e. when the decomposition exists, the set of solutions usually contains a small number of points, actually exactly one if the first part of the above conjecture holds. In this case, the ideal is maximal and its grevlex Gröbner basis contains only linear polynomials; recovering the solution is then immediate. If the ideal is zero-dimensional but not of degree 1, it is still simple to recover the solutions since the degree is small. Exceptionally, e.g. if R decomposes as a sum of $n-3$ points, the dimension of the fiber is positive; this can be easily detected on the grevlex basis. It is then still possible to deduce some useful relations, but one can also simply discard this rarely occurring decomposition trial. This situation is in stark contrast with the one in Gaudry and Diem's algorithm, where we have seen that the degree of the ideal is generically equal to the Bézout bound $2^{n(n-1)}$. This means that the solution set generically contains $2^{n(n-1)}$ points, although most of them lie in an extension of $\mathbb{F}_q$ since the probability of finding a decomposition is only $1/n!$. In their setting, the computation of a degrevlex Gröbner basis is thus not sufficient to solve the system and the FGLM algorithm is needed. We see below that this over-exponential degree in n is actually the bottleneck of their approach.

In order to derive effective upper bounds for the complexity of the grevlex Gröbner basis computation of a zero-dimension system $\{f_1, \ldots, f_r\}$, it is necessary to make some additional hypotheses. For instance, one can assume that the sequence $\{f_1, \ldots, f_r\}$ is semi-regular [1, 2] or that the set of solutions of the homogenized system has no positive dimension component at infinity [32]. These properties hold generically, and imply that the maximum degree of polynomials occurring during the computation of the Gröbner basis is bounded by the degree of regularity d_{reg} of the homogenized system, which is itself smaller than the Macaulay bound $\sum_{i=1}^{r}(\deg f_i - 1) + 1$. The standard algorithms for Gröbner bases (e.g. Buchberger [7], Faugère's F4 and F5 [15, 16]) can then be reduced to the computation of the row echelon form of the d_{reg}-Macaulay matrix (cf [32]). In the following, we make a second assumption, which has been verified in all our experiments:

Assumption 2 *The maximal degree of the polynomials occurring during the computation of the homogenized grevlex Gröbner basis of the system $\{\varphi_0, \ldots, \varphi_{n-1}\}$ arising from (5) is smaller than the Macaulay bound $d = \sum_{i=0}^{n-1}(\deg \varphi_i - 1) + 1$.*

Using the fact that the system in our variant is composed of polynomials of degree 2^{n-2} in $n-1$ variables, we obtain that $d = n2^{n-2} - n + 1$. The number of columns of the d-Macaulay matrix is at most the number of monomials of degree smaller than or equal to d, which in our case is bounded by $\binom{n2^{n-2}}{n-1}$. Similarly, the number of rows is less than $n\binom{(n-1)2^{n-2}}{n-1}$, corresponding to the multiples up to degree d of the n initial

polynomials. Since there are more columns than rows, we obtain with fast reduction techniques the following bound:

$$\tilde{c}(n,q) = \tilde{O}\left(\binom{n2^{n-2}}{n-1}^{\omega}\right),$$

where $\omega \leq 3$ is the effective complexity exponent of matrix multiplication (in practice, $\omega = \log_2(7)$ when using Strassen's multiplication). As $n-1$ is negligible compared to $n2^{n-2}$, using Stirling's formula we get

$$\binom{n2^{n-2}}{n-1} \sim \frac{(n2^{n-2})^{n-1}}{(n-1)!} \sim 2^{(n-1)(n-2)} e^n \, (2\pi n)^{-1/2}.$$

This directly implies our main result:

Theorem 1 *Let E be an elliptic curve defined over $\mathbb{F}_{q^n}$ and G a cyclic subgroup of its group of rational points. If Assumptions 1 and 2 are satisfied, then the DLP in G can be solved with asymptotic complexity*

$$\tilde{O}\left((n-1)!\left(2^{(n-1)(n-2)} e^n \, n^{-1/2}\right)^{\omega} q^2\right). \tag{7}$$

Note however that the upper-bound given for $\tilde{c}(n,q)$ (and thus the estimate of Theorem 1) is convenient but not sharp: it does not take into account the fact that the Macaulay matrix is sparse and heavily structured. Actually, deriving precise bounds for the computation of Gröbner bases is still an open problem. As an illustration, for $n = 5$ the above estimate predicts about 10^{14} multiplications in $\mathbb{F}_q$ to achieve the Gröbner basis computation, whereas the experimental computation with F4 only requires about 10^{10} multiplications. We see in section 4 how to reduce this last amount by a factor 3.

In the same spirit, we can also try to improve the estimate of the complexity of Gaudry and Diem's algorithm. We have seen that the resolution of the polynomial system is composed of two main stages: the computation of a grevlex Gröbner basis followed by the ordering change algorithm FGLM. The cost of the first step can be roughly estimated in the same way as for our variant and is in $\tilde{O}\left(\binom{n2^{n-1}+1}{n}^{\omega}\right) = \tilde{O}\left(\left(2^{n(n-1)} e^n \, n^{-1/2}\right)^{\omega}\right)$. As already mentioned, the complexity of FGLM for a zero-dimensional ideal of degree D in n variables is $\tilde{O}(nD^3)$, and this estimate is actually sharp. Consequently, for $\omega < 3$ we find that the cost of the lexicographic Gröbner basis computation is dominated by the cost of FGLM. Its complexity is in $\tilde{O}\left((2^{n(n-1)})^3\right)$, and thus the total complexity of Gaudry and Diem's version is $\tilde{O}\left(n! 2^{3n(n-1)} q^{2-2/n}\right)$. An easy computation then shows that our approach is asymptotically faster[1], provided $n \geq \sqrt[3]{\left(\frac{2}{3-\omega} + \epsilon\right) \log_2 q}$.

Similarly, our method is asymptotically faster than Pollard rho algorithm if $n \leq \left(\frac{1}{2\omega} - \epsilon\right) \log_2 q$. We stress that these comparisons are only asymptotic; we have seen before that our version is irrelevant if $n \leq 4$. However, this shows that for $n > 4$, there exists a range of values for q in which our algorithm is the most efficient.

2.3 Comparison with the hybrid approach

We have seen that the main difficulty in Gaudry and Diem's algorithm is the resolution of the polynomial system. Recently, Bettale et al. [4] have proposed a hybrid approach for solving such systems: the idea is

[1] Some papers (e.g. [18]) claim that in some special cases and with some modifications, the complexity exponent of FGLM is actually smaller than 3. If the Gröbner basis computation dominates the cost of the resolution, then the comparison is somewhat different: our approach would be asymptotically faster than Gaudry and Diem's for $n \geq \sqrt{\left(\frac{1}{\omega} + \epsilon\right) \log_2 q}$. Nevertheless, in our experiments the FGLM step was always the longest by a large margin.

to find a solution by exhaustive search on some variables and Gröbner basis computations of the modified systems where the selected variables have been *specialized* (i.e. evaluated). It is thus a trade-off between exhaustive search and Gröbner basis techniques. A natural choice here would be to specialize (or guess) one variable. The exhaustive search multiplies by q the number of polynomial systems, but these systems now consist of n equations in $n-1$ variables. At first sight, this seems quite similar to our version; however, the total degree of the equations in this hybrid approach is 2^{n-1} whereas it is only 2^{n-2} in our case. The following chart summarizes the number of multivariate systems to solve together with their parameters, in order to find one relation in $E(\mathbb{F}_{q^n})$. It shows that our version provides a better trade-off between the number of systems to solve and their complexity than the hybrid approach.

Method	average number of systems	number of equations	number of variables	total degree
Gaudry-Diem	$n!$	n	n	2^{n-1}
Gaudry-Diem with hybrid approach	$n!\,q$	n	$n-1$	2^{n-1}
this work	$(n-1)!\,q$	n	$n-1$	2^{n-2}

2.4 Application to $\mathbb{F}_{q^5}$

The approach of Gaudry and Diem, while theoretically interesting, turns out to be intractable on $\mathbb{F}_{q^n}$ as soon as $n \geq 5$. Not only is the computation of the 6-th summation polynomial problematic (cf. section 3), but also, since the system arising from (3) has a large number of solutions (about $2^{5(5-1)} \simeq 10^6$) in $\overline{\mathbb{F}_{q^5}}$, it is very difficult to solve. Indeed, we remind that the complexity of the resolution (e.g. by using FGLM to obtain a lex order Gröbner basis) depends of the degree of the ideal generated by the equations, which is generically $2^{n(n-1)}$. A natural way of decreasing this degree would be to add the field equations $e_i^q - e_i$, but clearly this is not practical for large values of q. In particular, we have not been able to successfully run one complete relation search with their method, as the requested memory exceeded the capacity of our personal computer. Nonetheless, using our algorithm and our own implementation of the F4 variant (see section 4), we are able to check and if necessary compute a decomposition over $\mathbb{F}_{p^5}$ with p a prime number of 32 bits in about 8.5 sec on a 2.6 GHz Intel Core 2 Duo processor.

In characteristic 2, the computation is much faster: as pointed out by Granger [25], Semaev's summation polynomials are sparser than in the odd characteristic case, so that the corresponding systems are much easier to solve. Note however that the bounds given for the degree of regularity and the degree of the ideals remain the same, as do the complexity estimates. Timings for testing decompositions up to $n = 4$ with Gaudry-Diem approach are given in [25], but the simplification provided by the characteristic 2 case is still not sufficient to make this approach work for $n = 5$ on a personal computer. Interestingly, the speedup noticed by Granger also applies with our method. Indeed, in characteristic 2, testing a decomposition in four points over $\mathbb{F}_{2^{160}} = \mathbb{F}_{(2^{32})^5}$ takes only 30 ms instead of the 8.5 sec given above in large characteristic.

Unfortunately, this is still much too slow to yield in a reasonable time the solution of the ECDLP over fields of size compatible with current levels of security. To be more precise, we can estimate for which base fields our algorithm is faster than Pollard rho, knowing that a single decomposition test requires about 3.10^9 multiplications in $\mathbb{F}_p$ for p odd and about 2.10^7 in $\mathbb{F}_{2^d}$ for the binary case (since the relation search is the dominating step, we can neglect the linear algebra part). Our method is then faster as soon as the cardinality of the base field is greater than 2^{60} in the odd characteristic case, or 2^{45} in characteristic 2.

Nevertheless, our approach provides an efficient attack of non-standard problems such as the oracle-assisted static Diffie-Hellman problem, as explained in section 5.

3 Computing the symmetrized summation polynomials

The main difficulty of the previously investigated algorithms is the construction of relations of the form (2) or (5). Semaev's summation polynomials were first proposed in [41] to solve, or at least, to reduce this difficulty, allowing to translate this problem into the resolution of the polynomial equation $f_m(x_{P_1}, \ldots, x_{P_{m-1}}, x_R) = 0$, where $x_{P_1}, \ldots, x_{P_{m-1}}$ are the unknowns. The m-th polynomial f_m is computed only once and is evaluated in x_R for each relation search. As mentioned above, it is more efficient to express this equation in terms of the elementary symmetric functions of the unknowns

$$e_1 = \sum_i x_{P_i}, \quad e_2 = \sum_{i<j} x_{P_i} x_{P_j}, \quad \ldots \quad , \quad e_{m-1} = \prod_i x_{P_i},$$

before the resolution of the system. This symmetrizing operation greatly reduces the total degree of the system, and improves a lot its resolution by e.g. Gröbner basis techniques. It can be done once for all at the beginning of the relation search.

We propose here two distinct improvements: both consider a direct computation of the symmetrized summation's polynomials, instead of rewriting the equation $f_m(x_{P_1}, \ldots, x_{P_{m-1}}, x_R) = 0$ in terms of elementary symmetric polynomials after the computation of Semaev's polynomials, as in [21]. Hence for $m = 5$, the first method allows to reduce the computation time and the memory requirement by a factor almost equal to 10; while the second technique seems to provide a less significant advantage in term of computation time, it allows to reduce the memory requirement by a factor 25 (on Magma V2.17-5 [5]).

3.1 Distributing the symmetrization

Let us recall that in [41], the summation polynomials are determined recursively, each inductive step consisting of a resultant computation. Our first improvement is to partially symmetrize after each step: it has the double benefit of reducing the size of the intermediate polynomials and the cost of the final symmetrization, by distributing it between the different steps. It is summarized by the following proposition:

Proposition 5 *Let E be an elliptic curve defined over a field K of characteristic different from 2 or 3, with reduced Weierstrass equation $y^2 = x^3 + ax + b$. The symmetrized summation polynomials are determined by the following induction. The initial value for $n = 3$ is given by*

$$\tilde{f}_3(e_{1,2}, e_{2,2}, X_3) = \left(e_{1,2}^2 - 4e_{2,2}\right) X_3^2 - 2\left(e_{1,2}(e_{2,2} + a) + 2b\right) X_3 + (e_{2,2} - a)^2 - 4b\, e_{1,2}$$

and for $m \geq 3$ by

$$\tilde{f}_{m+1}(e_{1,m}\,, \ldots, e_{m,m}\,, X_{m+1}) = Sym_m\left(Res_Y\left(\tilde{f}_m(e_{1,m-1}, \ldots, e_{m-1,m-1}, Y), f_3(e_{1,1}, X_{m+1}, Y)\right)\right),$$

where

- *$e_{r,n}$ is the r-th elementary symmetric polynomial in variables $X_1, \ldots, X_n$,*
- *$f_3(X_1, X_2, X_3) = (X_1 - X_2)^2 X_3^2 - 2\left((X_1 + X_2)(X_1X_2 + a) + 2b\right) X_3 + (X_1X_2 - a)^2 - 4b(X_1 + X_2)$ is Semaev's third summation polynomial,*
- *Sym_m denotes the operation of rewriting a partially symmetrized polynomial in terms of elementary symmetric functions*

$$\begin{cases} e_{1,m} = e_{1,1} + e_{1,m-1} \\ e_{2,m} = e_{1,1}e_{1,m-1} + e_{2,m-1} \\ \qquad \vdots \\ e_{m-1,m} = e_{1,1}e_{m-2,m} + e_{m-1,m-1} \\ e_{m,m} = e_{1,1}e_{m-1,m-1} \end{cases}$$

Obviously it is also possible to define $\tilde{f}_m$ from resultants of $\tilde{f}_{m-j}$ and $\tilde{f}_{j+2}$ for $1 \leq j \leq m-3$, as in [41]. This has the advantage of reducing the number of resultant computations, but increases the complexity of the symmetrization step. In our context, since m is small ($m \leq 6$), the approach of Proposition 5 is the fastest.

3.2 Divisors and elimination

In our second improvement, we replace the computation of resultants and the symmetrization by an elimination order Gröbner basis computation. Let E be an elliptic curve of equation $y^2 = x^3 + ax + b$ defined over K of characteristic different from 2 or 3. We consider the principal divisor $D = (P_1) + \ldots + (P_m) - m(\mathcal{O}_E) \in Div^0_K(E)$ where $P_1, \ldots, P_m \in E(K)$ are such that $P_1 + \ldots + P_m = \mathcal{O}_E$. Up to a constant, there exists a unique function $g_m \in K(E)$ such that $D = div(g_m)$ (cf. [43] Corollary 3.5). The same techniques as the ones used in Miller's algorithm [35] enable us to express the function g_m.

Let $l_i(X, Y) = 0$ $(1 \leq i \leq m-1)$ be the equations of the lines passing through $P_1 + \ldots + P_i$ and P_{i+1} and $v_i(X, Y) = 0$ $(1 \leq i \leq m-2)$ the equations of the vertical lines passing through $P_1 + \ldots + P_{i+1}$, then we have

$$g_m(X, Y) = \frac{l_1 \ldots l_{m-1}}{v_1 \ldots v_{m-2}}(X, Y).$$

An easy induction shows that

$$g_m(X, Y) = g_{m,1}(X) + Y g_{m,2}(X) \tag{8}$$

where $g_{m,1}$ and $g_{m,2}$ are two polynomials of degree lower than $d_{m,1}$ and $d_{m,2}$ respectively:

$$d_{m,1} = \begin{cases} m/2 \text{ if } m \text{ is even} \\ (m-1)/2 \text{ if } m \text{ is odd} \end{cases} \quad \text{and } d_{m,2} = \begin{cases} (m-4)/2 \text{ if } m \text{ is even} \\ (m-3)/2 \text{ if } m \text{ is odd} \end{cases}$$

Note that the function g_m is uniquely determined if the equations of l_i and v_i are normalized at the point at infinity. The intersection between the curve $(g_m = 0)$ and E is exactly the set of points P_i, $1 \leq i \leq m$, thus the following proposition is quite direct:

Proposition 6 *Let E be an elliptic curve of equation $y^2 = x^3 + ax + b$ defined over a field K of characteristic different from 2 or 3. Let $P_1, \ldots, P_m \in E(K)$ be such that $P_1 + \ldots + P_m = \mathcal{O}_E$ and $g_{m,1}$ and $g_{m,2}$ be the polynomials given by equation (8). Then we have $g_{m,1}(x)^2 - (x^3 + ax + b) g_{m,2}(x)^2 = 0$ if and only if x is the x-coordinate of one of the points P_i.*
Conversely, if $g_{m,1}$ and $g_{m,2}$ are two arbitrary polynomials in $K[X]$ with degree $d_{m,1}$ and $d_{m,2}$, then the roots in $\overline{K}$ of

$$F_m(X) = g_{m,1}(X)^2 - (X^3 + aX + b)\, g_{m,2}(X)^2,$$

counted with multiplicity, are the x-coordinates of points $Q_1, \ldots, Q_m \in E(\overline{K})$ such that $Q_1 + \ldots + Q_m = \mathcal{O}_E$.

The second assertion comes from the fact that in $K(E)$, we have $F_m = (g_{m,1} + Y g_{m,2})(g_{m,1} - Y g_{m,2})$. Since $\deg(F_m) = m$, F_m has exactly m roots counted with multiplicity over $\overline{K}$, each of which is the x-coordinate of two opposite points $\pm Q_i \in E(\overline{K})$. Up to a change of sign, we can assume that Q_i is a zero of $g_{m,1} + Y g_{m,2}$ (and so $-Q_i$ is a zero of the second factor $g_{m,1} - Y g_{m,2}$). Thus, the principal divisor $Div(g_{m,1} + Y g_{m,2})$ is equal to $(Q_1) + \ldots + (Q_m) - m(\mathcal{O}_E)$, which implies $Q_1 + \ldots + Q_m = \mathcal{O}_E$.

We can now use this proposition to construct the symmetrized summation polynomials. Let $A = K[\alpha_0, \ldots, \alpha_{d_{m,1}}, \beta_0, \ldots, \beta_{d_{m,2}}, x_{P_1}, \ldots, x_{P_m}]$. We define the following elements of $A[X]$:

$$h_{m,1}(X) = \sum_{i=0}^{d_{m,1}} \alpha_i X^i, \quad h_{m,2}(X) = X^{d_{m,2}} + \sum_{i=0}^{d_{m,2}-1} \beta_i X^i,$$

$$F_m(X) = h_{m,1}(X)^2 - (X^3 + aX + b)\, h_{m,2}(X)^2$$

Finally, let I be the ideal of A generated by $F_m(x_{P_1}), \ldots, F_m(x_{P_m})$. We can easily find a different set of generators of I by identifying the coefficients of F_m with the elementary symmetric functions $e_1, \ldots, e_m$ of the variables $x_{P_1}, \ldots, x_{P_m}$, and consider the result as an ideal J of $K[\alpha_0, \ldots, \alpha_{d_{m,1}}, \beta_0, \ldots, \beta_{d_{m,2}}, e_1, \ldots, e_m]$. Elimination theory (cf. [10]) allows to compute efficiently (e.g. with appropriate Gröbner bases) a set of generators of the ideal $J' = J \cap K[e_1, \ldots, e_m]$. According to the second part of Proposition 6, a m-tuple $(e_1, \ldots, e_m)$ belongs to the algebraic set $\mathbf{V}(J')$ if and only if the roots of the polynomial $T^m + \sum_{i=1}^{m} (-1)^i e_i T^{m-i}$ are the x-coordinates of points of $E(\overline{K})$ whose sum is the point at infinity $\mathcal{O}_E$. Actually, using Semaev's results, this elimination ideal J' is principal, generated by the m-th symmetrized summation polynomial. Hence, this elimination computes the m-th summation polynomial directly in terms of $e_1, \ldots, e_m$. Note that with this approach, it is also possible to compute directly the partially symmetrized polynomial $\tilde{f}_m$.

A worked example

For $m = 5$, following the previous construction, we obtain

$$F_5(X) = (\alpha_2 X^2 + \alpha_1 X + \alpha_0)^2 - (X^3 + aX + b)(X + \beta_0)^2$$

with $a, b \in K$. By identifying the coefficients of this polynomial with the elementary symmetric polynomials $e_1, \ldots, e_5$ of the variables $x_{P_1}, \ldots, x_{P_4}, x_{P_5}$, we deduce the polynomial system

$$\begin{cases} e_1 = \alpha_2^2 - 2\beta_0 \\ e_2 = \beta_0^2 + a - 2\alpha_1\alpha_2 \\ e_3 = 2\alpha_0\alpha_2 + \alpha_1^2 \\ e_4 = a\beta_0^2 + 2b\beta_0 - 2\alpha_0\alpha_1 \\ e_5 = \alpha_0^2 - b\beta_0^2 \end{cases}$$

and using a Gröbner basis computation with an elimination order, we obtain the fifth summation polynomial f_5 directly in terms of $e_1, \ldots, e_5$.

3.3 Some comparisons

Here we give a comparison of computer times between the classical computation using resultants followed by a final symmetrization and our two methods. In all cases, we have used the software Magma (V2.17-5) on a 2.6 GHz Intel Core 2 processor; the symmetrizations have been done via an elimination order Gröbner basis computation. In view of the applications we have in mind, we chose to compute the 5-th symmetrized summation polynomial on an extension field $\mathbb{F}_{p^5}$, p prime.

$\log_2(p)$	resultant + symmetrization	1st method	2nd method
8	$1.54 + 10.45 = 11.99$ sec	1.04 sec	1.75 sec
16	$1.58 + 10.63 = 12.21$ sec	1.04 sec	1.77 sec
32	$10.23 + 23.16 = 33.39$ sec	3.57 sec	11.12 sec
memory requirement	510 MB	66 MB	22 MB

We also tried to perform the same computations for the 6-th symmetrized summation polynomial. Unfortunately, in this case, both resultant and Gröbner based computations exceeded the memory capacity of our personal computer (about 4 GB). However, we were able to obtain with our first method (partially

symmetrized resultants) the 6-th symmetrized summation polynomial over $\mathbb{F}_{p^6}$ ($|p|_2 = 27$), for a fixed value of the last variable x_6, in about 10 min using 60 MB, expressed in the variables $e_{1,5}, \ldots, e_{5,5}$. In practice, this would mean that each time we try a decomposition of a new point into 5 points of the factor base, we would have to pay this extra price of computing the corresponding 6-th symmetrized summation polynomial. Clearly, this would slow down unreasonably any of the techniques presented in this paper.

3.4 The characteristic 2 case

The previous results can be easily adapted in characteristic 2. We consider an ordinary elliptic curve E defined over $\mathbb{F}_{2^d}$, with reduced Weierstrass equation $y^2 + xy = x^3 + ax^2 + b$. Then the third Semaev's summation polynomial is

$$f_3(X_1, X_2, X_3) = (X_1X_2 + X_1X_3 + X_2X_3)^2 + X_1X_2X_3 + b,$$

see [41], and its partial symmetrization is

$$\tilde{f}_3(e_{1,2}, e_{2,2}, X_3) = (e_{1,2}X_3 + e_{2,2})^2 + e_{2,2}X_3 + b.$$

We can then proceed as in Proposition 5 to compute $\tilde{f}_m$. For the second method, one just has to replace $g_{m,1}(X)^2 - g_{m,2}(X)^2(X^3 + aX + b)$ by $g_{m,1}(X)^2 + Xg_{m,1}(X)g_{m,2}(X) + (X^3 + aX^2 + b)g_{m,2}(X)^2$.

As already mentioned in section 2.4, the summation polynomials are much sparser in characteristic 2, and are thus faster to compute. For instance, the 5-th partially symmetrized summation polynomial has only 100 terms and its computation takes about 50 ms with our first method over $\mathbb{F}_{(2^{31})^5}$, whereas in the odd characteristic case, it has 3972 terms and is computed in 4 sec over $\mathbb{F}_{p^5}$ for a prime p of comparable size.

4 An F4-like algorithm without reduction to zero

An efficient way to solve the multivariate polynomial system coming from (2) or (5) is to use Gröbner basis tools. Currently, the best algorithms for constructing Gröbner bases are Faugère's F4 and F5 [15, 16], which are improvements of the classical Buchberger's algorithm. The second one, F5, is considered as the most efficient, since it includes a criterion to eliminate a priori almost all critical pairs that eventually reduce to zero. This criterion is based on the concept of "signature" of a polynomial; the main drawback is that many reductions are forbidden because they do not respect signature compatibility conditions. Hence, the polynomials considered in the course of the F5 algorithm are mostly "top-reduced" but their tails are left almost unreduced; this increases significantly the complexity of the remaining pairs' reduction. Furthermore, F5 generates many "redundant" polynomials, i.e. which are not members of a minimal Gröbner basis, but cannot be discarded for signature reasons [13]. The total number of computed critical pairs thus remains relatively important, at least compared to what could be expected from the F4 algorithm if all critical pairs reducing to zero were removed. These drawbacks are especially significant for overdetermined systems such as those we are considering. As mentioned by Faugère in [16], this is a consequence of the incremental nature of the F5 algorithm. Indeed, to determine a Gröbner basis of an ideal generated by m polynomials, F5 starts by computing a basis of the ideal generated by the first $m-1$ polynomials. Clearly, the additional equation of the overdetermined system cannot provide any speed-up at this point. Moreover, in our case, since the systems considered during the relation search always have the same shape, it is possible to extract from a precomputation the knowledge of the relevant critical pairs and to remove the pairs that lead to zero reductions. When such a precomputation is accessible, there is no reason to use F5 instead of F4.

Recall that during the course of the F4 algorithm, a queue of yet untreated critical pairs is maintained. At each iteration of the main loop, some pairs are selected from this queue (according to some predefined strategy,

usually all pairs having the smallest lcm total degree) and treated, that is, their S-polynomials are computed and reduced simultaneously using linear algebra tools and results of former computations. The queue is then updated with the critical pairs involving the resulting new generators and satisfying Buchberger's first and second criteria [7, 24]. Here is a quick outline of the method we used for our computations:

1. For precomputation purposes, run a standard F4 algorithm on the first system, with the following modifications:
   - At each iteration, store the list of all selected critical pairs.
   - Each time there is a reduction to zero, remove from the stored list the critical pair that leads to the reduction.
2. For each subsequent system, run a F4 computation with the following modifications:
   - Do not maintain nor update a queue of untreated pairs.
   - At each iteration, instead of selecting pairs from the queue, pick directly from the previously stored list the relevant pairs.

More details on this method as well as applications to other problems are given in [29].

This algorithm is probabilistic since the precomputation is not necessarily compatible with all the following systems. Fortunately, one can always detect when a subsequent system behaves "non-generically", and then resume the computation with the classical F4 algorithm. An upper-bound for the probability of failure is given in [29, Thm. 4] under some worst-case hypotheses. Assuming that the precomputation has been done in n_{step} iterations, the variant of F4 computes the Gröbner basis of a system involved in the decomposition step, with a probability heuristically greater than $c(p)^{n_{step}}$, where $\mathbb{F}_p$ is the base field and $c(p) = 1 - 1/p + O(1/p^2)$. In particular, when p is large, the probability of failure is very close to 0.

As an illustration of this approach we give some examples of the speed gain it provides on $\mathbb{F}_{p^5}$, using the equations generated from the fifth summation polynomial. The system to solve is composed of 5 equations defined over $\mathbb{F}_p$ of total degree 8 in 4 variables. We run a degrevlex Gröbner basis computation of the corresponding ideal over four prime fields of sizes 8, 16, 25 and 32 bits. To be fair, we compare our variant `F4Remake` with an implementation of F4 which uses the same primitives and structures (in language C), and also with the proprietary software Magma (V2.15-15) whose implementation is probably the best publicly available for the considered finite fields. All tests are performed on a 2.6 GHz Intel Core 2 Duo processor, the timings are given in seconds. We give the estimated probabilities of failure, thus showing that `F4Remake` succeeds for almost all systems.

size of p	est. failure probability	F4Precomp	F4Remake	F4	F4/F4Remake	F4 Magma
8 bits	0.11	8.963	2.844	5.903	2.1	9.660
16 bits	4.4×10^{-4}	(19.07)	3.990	9.758	2.4	9.870
25 bits	2.4×10^{-6}	(32.98)	4.942	16.77	3.4	118.8
32 bits	5.8×10^{-9}	(44.33)	8.444	24.56	2.9	1046

Step	degree	F4Remake matrix size	F4 matrix size	size ratio
14	17	1062×3072	1597×3207	1.6
15	16	1048×2798	1853×2999	1.9
16	15	992×2462	2001×2711	2.2
17	14	903×2093	2019×2369	2.5
18	13	794×1720	1930×2000	2.8

Fig. 2. Experimental results on $E(\mathbb{F}_{p^5})$

The `F4Remake` algorithm requires a single precomputation of 8.963 sec to generate the list of relevant pairs. The above timings show that this overhead is largely compensated as soon as there are more than two subsequent computations. We emphasize that this precomputed list of relevant pairs is the same for the four cases $|p|_2 = 8, 16,\ 25$ or 32 bits. We have also solved this system with our own implementation of the F5 algorithm[1]. The size of the Gröbner basis computed by F5 at the last step (before minimization) is surprisingly large: it contains 17249 labeled polynomials whereas both versions of F4 never build more than 2789 polynomials at once, and construct bases containing at most 329 generators. Note that these figures do not depend on the implementation's details. The large number of polynomials that F5 computes has obvious consequences on its performances; in particular, the timings of F5 that we have obtained for this system are much worse than those of F4 or its variants. This shows that F5 as described in [16] is unsuitable for these specific systems.

5 Application to an oracle-assisted static Diffie-Hellman algorithm

Semaev's idea of decomposing points of $E(\mathbb{F}_{q^n})$ into a well-suited factor base leads naturally to an oracle-assisted resolution of the SDHP, akin to the finite field SDHP algorithm presented in [28]. We first recall here the definition of oracle-assisted SDHP from [6]:

Definition 7 *Let G be a finite group of order $\#G$ and $P, Q \in G$ such that $Q = [d]P$ where $d \in [1, \#G - 1]$ is a secret integer. An algorithm $\mathcal{A}$ is said to solve the SDHP in G if, given P, Q, and a challenge $X \in G$, it outputs $[d]X \in G$.*
The SDHP-solving algorithm $\mathcal{A}$ is said to be oracle-assisted if, during a learning phase, it can make any number of queries $X_1, \ldots, X_l$ to an oracle that outputs $[d]X_1, \ldots, [d]X_l$, after which $\mathcal{A}$ is given a previously unseen challenge X and outputs $[d]X$.

Generally, the ability to decompose points into a factor base $\mathcal{F} = \{P_1, \ldots, P_l\}$ gives the following oracle-assisted algorithm:

- learning phase: ask the oracle to compute $Q_i = [d]P_i$ for $1 \leq i \leq l$,
- decompose a challenge X as $X = \sum_i [c_i]P_i$ and answer $Y = \sum_i [c_i]Q_i$.

This methodology directly applies to the case $G = E(\mathbb{F}_{q^n})$ with the factor base $\mathcal{F} = \{P \in E(\mathbb{F}_{q^n}) : P = (x_P, y_P),\ x_P \in \mathbb{F}_q,\ y_P \in S\}$. The only minor difficulty is that a small fraction of points actually decompose (1 in $n!$ or $q\,(n-1)!$ depending of the details). However, we can use a simple variation of the descent step to circumvent this difficulty:

1. learning phase: ask the oracle to compute $Q = [d]P$ for each $P \in \mathcal{F}$
2. descent: given a challenge X, pick a random integer r coprime to the order of G and compute $X_r = [r]X$
3. check if X_r can be written as a sum of m points of $\mathcal{F}$: $X_r = \sum_{i=1}^{m} \epsilon_i P_i$, with $\epsilon_i \in \{-1; 1\}$
4. if X_r is not decomposable, go back to step 2; else output $Y = [s]\,(\sum_{i=1}^{m} \epsilon_i Q_i)$ where $s = r^{-1} \mod \#G$.

It should be noted that the same technique can also be used to solve other variants of SDHP, such as the "Delayed Target" Discrete Logarithm or Diffie-Hellman problem (DTDLP and DTDHP) described in [31]. Note also that a similar approach has been presented independently by Granger in [25].

To our knowledge, the only other known method for solving the SDHP over a general elliptic curve consists in solving the underlying discrete logarithm problem, i.e. computing d given P and $Q = [d]P$. It is not obvious to compare this with our technique because the cost of the learning phase is difficult to estimate. But if the definition field is large enough, for instance if E is an elliptic curve defined over $\mathbb{F}_{2^{155}}$, then generic methods like Pollard rho are currently unable to achieve the resolution of the DLP on E. However, testing

[1] At the present time, we have found no public implementation of F5 which achieves the computation of the complete Gröbner basis in a reasonable time.

a decomposition on such a curve takes only 22.95 ms on a 2.93 GHz Intel Xeon processor. This means that it would take less than 2 weeks to find a decomposition with 1000 of the above processors (after a learning phase of 2^{30} oracle queries), and shows that a complete attack on the oracle-assisted SDHP can be realized on $E(\mathbb{F}_{2^{155}})$ [26].

In practice, the oracle is often limited to a single device, e.g. a smart card chip, whereas the decomposition tests can be distributed over several powerful computers. Thus, the oracle queries are clearly the bottleneck of this SDHP resolution. As explained in [31], it is possible to rebalance the two steps by artificially reducing the factor base to a subset $\mathcal{F}'$ of cardinality $(\#\mathcal{F})/l$, where $l > 1$. This decreases the number of oracle calls by a factor l, and increases the complexity of the second step (more accurately, the expected number of trials before finding a decomposition) by a factor l^m, where m is the number of points in the decompositions. The optimal trade-off depends of the oracle and the computing power available. Note that since no linear algebra is done, this rebalancing is much simpler than the one or double large prime variation.

Using the estimates of section 2.2, we can compare our version with Gaudry-Diem's for the oracle-assisted SDHP on $E(\mathbb{F}_{q^n})$. For simplicity, we choose the same reducing factor l for both approaches, so that the number of oracle calls is $q/(2l)$ in both cases. The complexity of the decomposition step is $n!\, l^n\, c(n,q)$ with Gaudry-Diem method vs $(n-1)!\, l^{n-1}\, q\, \tilde{c}(n,q)$ with ours. An easy computation shows that asymptotically, our variant is better as soon as $n \geq \sqrt{\frac{1}{3-\omega}\log_2(q/l)}$, corresponding to a somewhat smaller range of values than for the ECDLP, but becoming larger as one wants to lessen the number of oracle calls.

In practice, we have seen that, with current personal computers, we can only decompose a given point into at most four points of the factor base. For $m = 4$ on $E(\mathbb{F}_Q)$, the complexity obtained by reducing the factor base is $\tilde{O}(Q^{1/5})$ if $Q = q^4$, which is the same complexity as our method when $Q = \tilde{q}^5$; however, the hidden constant is much smaller in our case. Indeed, we can detail the computation in both cases:

	$Q = q^4$	$Q = \tilde{q}^5$ [this work]
nb of oracle calls	$\frac{Q^{1/4}}{2l}$	$\frac{Q^{1/5}}{2\tilde{l}}$
decomposition cost	$4!\, l^4\, c(4, Q^{1/4})$	$4!\, \tilde{l}^4\, Q^{1/5}\, \tilde{c}(4, Q^{1/5})$

For a fair comparison, we equate the number of oracle calls, i.e. we choose $\tilde{l} = lQ^{1/5}/Q^{1/4} = lQ^{-1/20}$. With this choice, the cost of decomposition in our approach becomes $4!\, l^4 \tilde{c}(4, Q^{1/5})$. As explained in section 2.2, $\tilde{c}(4, Q^{1/5}) < c(3, Q^{1/5}) \ll c(4, Q^{1/4})$; for instance, for $Q = 257^{5\times 4}$, we found $\tilde{c}(4, Q^{1/5}) = 768$ sec and $c(4, Q^{1/4}) = 15476$ sec using Magma (V2.15-15). Similarly, in characteristic 2, letting $Q = 2^{160}$, we find $\tilde{c}(4, Q^{1/5}) = 0.67$ sec and $c(4, Q^{1/4}) = 272$ sec. In both cases, we see that, for a given field size, the oracle-assisted elliptic curve SDHP is easier over degree 5 than over degree 4 extension fields.

6 Conclusion and perspectives

In this article, we have shown that considering decomposition of points on $E(\mathbb{F}_{q^n})$ as sums of $n-1$ points improves the index calculus proposed by [11, 21], when $n \geq 5$ and $\log_2 q \leq O(n^3)$. The key point of our approach is that such decompositions lead to overdetermined polynomial systems, which are easier to solve than the systems arising from Gaudry and Diem's original version. This resolution can be greatly sped up by using our modified Gröbner basis computation algorithm, which takes advantage of the common shape of the systems and is therefore faster than F4 and F5. The complexity of our algorithm is still too large to seriously threaten ECDLP based cryptosystems with the current cryptographic key sizes. However, we

further illustrate the weakness of elliptic curves defined over small degree extension fields by providing an efficient way of solving oracle-assisted Diffie-Hellman problems.

For middle degree extension fields $\mathbb{F}_{q^n}$ with $n > 6$, the increased complexity of the decomposition tests means that none of the approaches presented in this paper can be realistically implemented. Instead of just reducing the number of points m relatively to n, a more natural choice would be to also enlarge the factor base $\mathcal{F}$ and consider the set of points in $E(\mathbb{F}_{q^n})$ whose x-coordinates lie in a $\mathbb{F}_q$-linear subspace of dimension $d > 1$. The probability that a point decomposes becomes approximately $\frac{q^{md-n}}{m!}$ and the linear algebra cost is $\tilde{O}(mq^{2d})$, so the best values for d and m involve a trade-off that depends heavily on the cost of the decomposition test. A major difficulty is that the equations given by the Weil restriction of Semaev's polynomials are no longer invariant under the full symmetric group $\mathfrak{S}_{md}$ acting on the variables, but only under the smaller group $\mathfrak{S}_m$. Besides, the unsymmetrized systems are too large to be directly solved. To take into account this $\mathfrak{S}_m$-invariance, a first idea, suggested by [9], is to work in the algebra associated to the invariant polynomial ring $\mathbb{F}_q\left[(X_{ij})_{1\leq i\leq m, 1\leq j\leq d}\right]^{\mathfrak{S}_m}$; but our experiments using the implementation provided by Magma have been unsuccessful so far. A more efficient way to solve these symmetric systems would probably be to use dedicated algorithms, such as SAGBI-Gröbner bases [46].

References

1. M. Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* PhD thesis, Université Pierre et Marie Curie, Paris VI, 2004.
2. M. Bardet, J.-C. Faugère, B. Salvy, and B.-Y. Yang. Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. Presented at MEGA'05, Eighth International Symposium on Effective Methods in Algebraic Geometry, 2005.
3. E. Becker, M. G. Marinari, T. Mora, and C. Traverso. The shape of the shape lemma. In *Proceedings of ISSAC'94*, pages 129–133, Oxford, 1994. ACM.
4. L. Bettale, J.-C. Faugère, and L. Perret. Hybrid approach for solving multivariate systems over finite fields. *J. Math. Cryptology*, 3(3):177–197, 2010.
5. W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).
6. D. R. L. Brown and R. P. Gallant. The static Diffie-Hellman problem. Cryptology ePrint Archive, Report 2004/306, 2004.
7. B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In N. Bose, editor, *Multidimensional systems theory, Progress, directions and open problems, Math. Appl. 16*, pages 184–232. D. Reidel Publ. Co., 1985.
8. H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren, editors. *Handbook of elliptic and hyperelliptic curve cryptography.* Discrete Mathematics and its Applications (Boca Raton). Chapman & Hall/CRC, Boca Raton, FL, 2006.
9. A. Colin. Solving a system of algebraic equations with symmetries. *J. Pure Appl. Algebra*, 117/118:195–215, 1997.
10. D. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms.* Undergraduate Texts in Mathematics. Springer, New York, third edition, 2007.
11. C. Diem. On the discrete logarithm problem in elliptic curves. *Compos. Math.*, 147(1):75–104, 2011.
12. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, IT-22(6):644–654, 1976.
13. C. Eder and J. Perry. F5C: a variant of Faugère's F5 algorithm with reduced Gröbner bases. *J. Symbolic Comput.*, 45(12):1442–1458, 2010.
14. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in cryptology - CRYPTO 1984*, volume 196 of *Lecture Notes in Comput. Sci.*, pages 10–18. Springer, Berlin, 1985.
15. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F4). *J. Pure Appl. Algebra*, 139(1-3):61–88, June 1999.
16. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *Proceedings of ISSAC'02*, pages 75–83, New York, NY, USA, 2002. ACM.

17. J.-C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *J. Symbolic Comput.*, 16(4):329–344, 1993.
18. J.-C. Faugère and L. Perret. Algebraic cryptanalysis of Curry and Flurry using correlated messages. In M. Yung and F. Bao, editors, *Inscrypt 2009*, volume 6151, pages 266–277, Berlin, Heidelberg, 2010. Springer-Verlag.
19. G. Frey and H.-G. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 62(206):865–874, 1994.
20. J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, Cambridge, second edition, 2003.
21. P. Gaudry. Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. *J. Symbolic Comput.*, 44(12):1690–1702, 2008.
22. P. Gaudry, F. Hess, and N. P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *J. Cryptology*, 15(1):19–46, 2002.
23. P. Gaudry, E. Thomé, N. Thériault, and C. Diem. A double large prime variation for small genus hyperelliptic index calculus. *Math. Comp.*, 76:475–492, 2007.
24. R. Gebauer and H. M. Möller. On an installation of Buchberger's algorithm. *J. Symbolic Comput.*, 6(2-3):275–286, 1988.
25. R. Granger. On the Static Diffie-Hellman Problem on elliptic curves over extension fields. In *Advances in cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Comput. Sci.*, pages 283–302, 2010.
26. R. Granger, A. Joux, and V. Vitse. New timings for oracle-assisted SDHP on the IPSEC Oakley 'Well Known Group' 3 curve. Announcement on the NBRTHRY mailing list, July 2010. `http://listserv.nodak.edu/archives/nmbrthry.html`.
27. F. Hess. Weil descent attacks. In *Advances in elliptic curve cryptography*, volume 317 of *London Math. Soc. Lecture Note Ser.*, pages 151–180. Cambridge Univ. Press, Cambridge, 2005.
28. A. Joux, R. Lercier, D. Naccache, and E. Thomé. Oracle assisted static Diffie–Hellman is easier than discrete logarithms. In M. G. Parker, editor, *IMA Int. Conf.*, volume 5921 of *Lecture Notes in Comput. Sci.*, pages 351–367. Springer, 2009.
29. A. Joux and V. Vitse. A variant of the F4 algorithm. In A. Kiayias, editor, *Topics in cryptology—CT-RSA 2011*, volume 6558 of *Lecture Notes in Comput. Sci.*, pages 356–375, Berlin, 2011. Springer.
30. N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.*, 48(177):203–209, 1987.
31. N. Koblitz and A. Menezes. Another look at non-standard discrete log and Diffie-Hellman problems. *J. Math. Cryptol.*, 2(4):311–326, 2008.
32. D. Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *Computer algebra (London, 1983)*, volume 162 of *Lecture Notes in Comput. Sci.*, pages 146–156. Springer, Berlin, 1983.
33. A. J. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. Inform. Theory*, 39(5):1639–1646, 1993.
34. V. S. Miller. Use of elliptic curves in cryptography. In *Advances in cryptology – CRYPTO 1985*, volume 218 of *Lecture Notes in Comput. Sci.*, pages 417–426. Springer, Berlin, 1986.
35. V. S. Miller. The Weil pairing, and its efficient calculation. *J. Cryptology*, 17(4):235–261, 2004.
36. S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Transactions on Information Theory*, IT-24:106–110, 1978.
37. J. M. Pollard. Monte Carlo methods for index computation (mod p). *Math. Comp.*, 32(143):918–924, 1978.
38. J. M. Pollard. Kangaroos, Monopoly and discrete logarithms. *J. Cryptology*, 13(4):437–447, 2000.
39. T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Comment. Math. Univ. St. Paul.*, 47(1):81–92, 1998.
40. I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Math. Comp.*, 67(221):353–356, 1998.
41. I. A. Semaev. Summation polynomials and the discrete logarithm problem on elliptic curves. Cryptology ePrint Archive, Report 2004/031, 2004.
42. D. Shanks. Class number, a theory of factorization, and genera. In *1969 Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Brook, N.Y., 1969)*, pages 415–440. Amer. Math. Soc., Providence, R.I., 1971.
43. J. H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1986.
44. N. P. Smart. The discrete logarithm problem on elliptic curves of trace one. *J. Cryptology*, 12(3):193–196, 1999.
45. N. Thériault. Index calculus attack for hyperelliptic curves of small genus. In Heidelberg, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Comput. Sci.*, pages 75–92. Springer, September 2003.

46. N. M. Thiéry. Computing minimal generating sets of invariant rings of permutation groups with SAGBI-Gröbner basis. In R. Cori, J. Mazoyer, M. Morvan, and R. Mosseri, editors, *DM-CCG 2001*, volume AA of *DMTCS Proceedings*, pages 315–328. Discrete Mathematics and Theoretical Computer Science, 2001.