Effective arithmetic in finite fields based on Chudnovsky's multiplication algorithm

Kévin Atighehchi, Stéphane Ballet, Alexis Bonnecaze, Robert Rolland

C. R. Acad. Sci. Paris, Ser. I ••• (••••) •••–•••

Contents lists available at ScienceDirect

C. R. Acad. Sci. Paris, Ser. I

www.sciencedirect.com

Number theory/Computer science

Effective arithmetic in finite fields based on Chudnovsky's multiplication algorithm

Arithmétique effective dans les corps finis basée sur l'algorithme de multiplication de Chudnovsky

Kévin Atighehchi [a], Stéphane Ballet [b], Alexis Bonnecaze [b], Robert Rolland [b]

[a] *Aix-Marseille Université, Laboratoire d'informatique fondamentale de Marseille, case 901, 13288 Marseille cedex 9, France*
[b] *Aix-Marseille Université, Institut de mathématiques de Marseille, case 930, 13288 Marseille cedex 9, France*

ARTICLE INFO

Article history:
Received 22 September 2015
Accepted after revision 1 December 2015
Available online xxxx

Presented by the Editorial Board

ABSTRACT

Thanks to a new construction of the Chudnovsky and Chudnovsky multiplication algorithm, we design efficient algorithms for both the exponentiation and the multiplication in finite fields. They are tailored to hardware implementation and they allow computations to be parallelized, while maintaining a low number of bilinear multiplications.

RÉSUMÉ

À partir d'une nouvelle construction de l'algorithme de multiplication de Chudnovsky et Chudnovsky, nous concevons des algorithmes efficaces pour la multiplication et l'exponentiation dans les corps finis. Ils sont adaptés à une implémentation matérielle et sont parallélisables, tout en gardant un nombre de multiplications bilinéaires très bas.

1. Introduction

Multiplication in finite fields is a fundamental operation in arithmetic and finding efficient multiplication methods remains a topical issue. Let q be a prime power, $\mathbb{F}_q$ the finite field with q elements and $\mathbb{F}_{q^n}$ the degree n extension of $\mathbb{F}_q$. If $\mathcal{B} = \{e_1, \ldots, e_n\}$ is a basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ then for $x = \sum_{i=1}^n x_i e_i$ and $y = \sum_{i=1}^n y_i e_i$, we have the product

$$z = xy = \sum_{h=1}^{n} z_h e_h = \sum_{h=1}^{n} \Bigg(\sum_{i,j=1}^{n} t_{ijh} x_i x_j \Bigg) e_h, \tag{1}$$

where $e_i e_j = \sum_{h=1}^n t_{ijh} e_h$, $t_{ijh} \in \mathbb{F}_q$ being some constants. The complexity of a multiplication algorithm in $\mathbb{F}_{q^n}$ depends on the number of multiplications and additions in $\mathbb{F}_q$. There exist two types of multiplications in $\mathbb{F}_q$: the scalar multiplication

E-mail addresses: kevin.atighehchi@univ-amu.fr (K. Atighehchi), stephane.ballet@univ-amu.fr (S. Ballet), alexis.bonnecaze@univ-amu.fr (A. Bonnecaze), robert.rolland@acrypta.fr (R. Rolland).

http://dx.doi.org/10.1016/j.crma.2015.12.001
1631-073X/

and the bilinear multiplication. The scalar multiplication is the multiplication by a constant (in $\mathbb{F}_q$) that does not depend on the elements of $\mathbb{F}_{q^n}$ that are multiplied. The bilinear multiplication is a multiplication of elements that depend on the elements of $\mathbb{F}_{q^n}$ that are multiplied. The bilinear complexity is independent of the chosen representation of the finite field. For example, the direct calculation of $z = (z_1, \ldots, z_n)$ using (1) requires n^2 non-scalar multiplications $x_i x_j$, n^3 scalar multiplications, and $n^3 - n$ additions.

More precisely, the multiplication of two elements of $\mathbb{F}_{q^n}$ is an $\mathbb{F}_q$-bilinear application from $\mathbb{F}_{q^n} \times \mathbb{F}_{q^n}$ onto $\mathbb{F}_{q^n}$. Then, it can be considered as an $\mathbb{F}_q$-linear application from the tensor product $\mathbb{F}_{q^n} \otimes_{\mathbb{F}_q} \mathbb{F}_{q^n}$ onto $\mathbb{F}_{q^n}$. Consequently, it can also be considered as an element T of ${\mathbb{F}_{q^n}}^\star \otimes_{\mathbb{F}_q} {\mathbb{F}_{q^n}}^\star \otimes_{\mathbb{F}_q} \mathbb{F}_{q^n}$, where $\star$ denotes the dual. Set

$$T = \sum_{i=1}^{r} x_i^\star \otimes y_i^\star \otimes c_i, \tag{2}$$

where the r elements $x_i^\star$ as well as the r elements $y_i^\star$ are in the dual ${\mathbb{F}_{q^n}}^\star$ of $\mathbb{F}_{q^n}$, while the r elements c_i are in $\mathbb{F}_{q^n}$. The following holds for any $x, y \in \mathbb{F}_{q^n}$: $x \cdot y = \sum_{i=1}^{r} x_i^\star(x) y_i^\star(y) c_i$. The decomposition (2) is not unique.

Definition 1.1. A bilinear multiplication algorithm $\mathcal{U}$ is an expression

$$x \cdot y = \sum_{i=1}^{r} x_i^\star(x) y_i^\star(y) c_i.$$

The number r of summands in this expression is called the bilinear complexity of the algorithm $\mathcal{U}$ and is denoted by $\mu(\mathcal{U})$.

Definition 1.2. The minimal number of summands in a decomposition of the tensor T of the multiplication is called the bilinear complexity of the multiplication and is denoted by $\mu_q(n)$:

$$\mu_q(n) = \min_{\mathcal{U}} \mu(\mathcal{U}),$$

where $\mathcal{U}$ is running over all bilinear multiplication algorithms in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.

The bilinear complexity of the multiplication in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ has been widely studied. In particular, it was proved in [2] that it is uniformly linear with respect to the degree n of the extension. This follows from the Chudnovsky and Chudnovsky multiplication algorithm (CCMA). This clever construction was originally introduced in 1987 in [3] and is based on the interpolation on algebraic curves.

There is benefit having a low bilinear complexity when considering hardware implementations mainly because it reduces the number of gates in the circuit. In this note, we consider three models.

- The non-scalar model (denoted NS), in which only the bilinear complexity is taken into account and it is assumed that all scalar operations are free. Indeed, this model does not reflect the reality and, since the bilinear complexity is not the whole complexity of the algorithm, the complexity of the linear part of the algorithm should also be taken into account.
- The model S1, which takes into account the number of multiplications without distinguishing between the bilinear ones and the scalar ones.
- The model S2, which takes into account all operations (multiplications and additions) in $\mathbb{F}_q$.

Notice that so far, practical implementations of multiplication algorithms over finite fields have failed to simultaneously optimize the number of scalar multiplications, additions, and bilinear multiplications.

Regarding exponentiation algorithms, the use of a normal basis is of interest because the qth power of an element is just a cyclic shift of its coordinates. A remaining question is how to implement multiplication efficiently in order to have simultaneously fast multiplication and fast exponentiation. In 2000, Gao et al. [6] showed that fast multiplication methods can be adapted to normal bases constructed with Gauss periods. They show that if $\mathbb{F}_{q^n}$ is represented by a normal basis over $\mathbb{F}_q$ generated by a Gauss period of type (n, k), the multiplication in $\mathbb{F}_{q^n}$ can be computed with $\mathrm{O}(nk \log nk \log\log nk)$ and the exponentiation with $\mathrm{O}(n^2 k \log k \log\log nk)$ operations in $\mathbb{F}_q$ (q being small). This result is valuable when k is bounded. However, in the general case, k is upper-bounded by $\mathrm{O}(n^3 \log^2 nq)$.

In 2009, Couveignes and Lercier constructed in [5, Theorem 4] two families of basis (called elliptic and normal elliptic) for finite field extensions, from which they obtained a model Ξ defined as follows. With every couple (q, n), they associated a model, $\Xi(q, n)$, of the degree-n extension of $\mathbb{F}_q$, such that the following holds: there is a positive constant K such that the following are true:

- elements in $\mathbb{F}_{q^n}$ are represented by vectors for which the number of components in $\mathbb{F}_q$ is upper bounded by $Kn(\log n)^2 \log(\log n)^2$;

– there exists an algorithm that multiplies two elements at the expense of $Kn(\log n)^4|\log(\log n)|^3$ multiplications in $\mathbb{F}_q$;
– exponentiation by q consists of a circular shift of the coordinates.

Therefore, for each extension of finite field, they show that there exists a model that allows both fast multiplication and fast application of the Frobenius automorphism. Their model has the advantage of existing for all extensions. However, the bilinear complexity of their algorithm is not competitive compared with the best known methods, as pointed out in [5, Section 4.3.4]. Indeed, it is clear that such a model requires at least $Kn(\log n)^2(\log(\log n))^2$ bilinear multiplications.

Note that here, the efficiency of the algorithms is described in terms of parallel time (depth of the circuit, in number of multiplications), number of processors (width), and total number of multiplications (size).

This article describes the main theoretical results of a more detailed forthcoming article, where an effective implementation for the case $\mathbb{F}_{16^{13}}$ is presented (for a preliminary version, see [1]).

2. New results

We propose another model with the following characteristics:

– our model is based on CCMA, thus the multiplication algorithm has a bilinear complexity in $O(n)$, which is optimal;
– our model is tailored to parallel computation. Hence, the computation time used to perform a multiplication or any exponentiation can easily be reduced with an adequate number of processors. Since our method has a bilinear complexity of multiplication in $O(n)$, it can be parallelized to obtain a constant time complexity using $\mathrm{O}(n)$ processors. The previous aforementioned works ([6] and [5]) do not give any parallel algorithm (such an algorithm is more difficult to conceive than a serial one);
– exponentiation by q is a circular shift of the coordinates and can be considered free. Thus, efficient parallelization can be done when doing exponentiation;
– the scalar complexity of our exponentiation algorithm is reduced, compare to a basic exponentiation using CCMA, thanks to a suitable basis representation of the Riemann–Roch space $\mathcal{L}(2D)$ in the second evaluation map. More precisely, the normal basis representation of the residue class field is carried in the associated Riemann–Roch space $\mathcal{L}(D)$, and the exponentiation by q consists of a circular shift of the n first coordinates of the vectors lying in the Riemann–Roch space $\mathcal{L}(2D)$;
– our model uses the Coppersmith–Winograd [4] method (denoted CW) or any variants thereof to improve matrix products and to diminish the number of scalar operations. This improvement is particularly efficient for exponentiation.

Theorem 2.1. *In the non-scalar model NS, there exist multiplication and exponentiation algorithms in $\mathbb{F}_{q^n}$ such that:*

– *the multiplication is done in parallel time in $\mathrm{O}(1)$ multiplications in $\mathbb{F}_q$ with $\mathrm{O}(n)$ processors, for a total in $\mathrm{O}(n)$ multiplications;*
– *exponentiation is done in parallel time in $\mathrm{O}(\log n)$ multiplications in $\mathbb{F}_q$ with $\mathrm{O}(n^2/\log^2 n)$ processors, for a total in $\mathrm{O}(n^2/\log n)$ multiplications.*

When considering models S1 and S2, two cases can be distinguished for the multiplication complexity. We might be interested either in the complexity of one multiplication or in the average (amortized) complexity of one multiplication when many multiplications are done simultaneously. Regarding exponentiation, a wise use of CW method allows complexity to be improved.

Theorem 2.2. *In the model S1, there exist multiplication and exponentiation algorithms in $\mathbb{F}_{q^n}$ such that:*

– *multiplication:*
  a) one multiplication is done in parallel time in $\mathrm{O}(1)$ multiplications in $\mathbb{F}_q$ with $\mathrm{O}(n^2)$ processors, for a total in $\mathrm{O}(n^2)$ multiplications;
  b) in the amortized sense, the parallel time is in $\mathrm{O}(1)$ multiplications in $\mathbb{F}_q$ with $\mathrm{O}(n^{1+\epsilon})$ processors, for a total in $\mathrm{O}(n^{1+\epsilon})$ multiplications where the value of ϵ is approximately 0.38 for the best known matrix product methods;
– *exponentiation is done in a parallel time of $\mathrm{O}(\log n)$ multiplications in $\mathbb{F}_q$ with $\mathrm{O}(n^{2+\epsilon}/\log^{2\epsilon} n)$ processors, for a total in $\mathrm{O}(n^{2+\epsilon}\log^{1-2\epsilon} n)$ multiplications.*

Theorem 2.3. *In the model S2, there exist multiplication and exponentiation algorithms in $\mathbb{F}_{q^n}$ such that:*

– *multiplication:*
  a) one multiplication is done in parallel time in $\mathrm{O}(\log n)$ operations in $\mathbb{F}_q$ with $\mathrm{O}(n^2/\log n)$ processors, for a total in $\mathrm{O}(n^2)$ operations;
  b) in the amortized sense, the parallel time is in $\mathrm{O}(\log n)$ operations in $\mathbb{F}_q$ with $\mathrm{O}(n^{1+\epsilon}/\log n)$ processors, for a total in $\mathrm{O}(n^{1+\epsilon})$ operations; recall that the value of ϵ is approximately 0.38 for the best matrix product methods;

Please cite this article in press as: K. Atighehchi et al., Effective arithmetic in finite fields based on Chudnovsky's multiplication algorithm, C. R. Acad. Sci. Paris, Ser. I (2016), http://dx.doi.org/10.1016/j.crma.2015.12.001

– *exponentiation is done in a parallel time of* $O(\log^2 n)$ *operations in* $\mathbb{F}_q$ *with* $O(n^{2+\epsilon}/\log^{1+2\epsilon} n)$ *processors, for a total in* $O(n^{2+\epsilon}\log^{1-2\epsilon} n)$ *operations.*

2.1. Multiplication and exponentiation algorithms

Let $F/\mathbb{F}_q$ be an algebraic function field over the finite field $\mathbb{F}_q$ of genus $g(F)$. We denote by $N_1(F/\mathbb{F}_q)$ the number of places of degree one of F over $\mathbb{F}_q$. If D is a divisor, $\mathcal{L}(D)$ denotes the Riemann–Roch space associated with D. We denote by F_Q the residue class field of the place Q which is isomorphic to $\mathbb{F}_{q^{\deg(Q)}}$, where $\deg(Q)$ is the degree of the place Q. The following theorem that makes effective the original algorithm groups some results of [2].

Theorem 2.4. *Let $F/\mathbb{F}_q$ be an algebraic function field of genus $g(F)$ defined over $\mathbb{F}_q$ and n an integer. Let us suppose that there exists a place Q of degree n.*

Then, if $N_1(F/\mathbb{F}_q) > 2n + 2g - 2$ there is an effective divisor D of degree $n + g - 1$ such that:

(i) Q is not in the support of D,
(ii) the evaluation map E defined by

$$\begin{aligned} E:\ \mathcal{L}(D) &\to F_Q \\ f &\mapsto f(Q) \end{aligned}$$

is an isomorphism of vector spaces over $\mathbb{F}_q$,
(iii) there exist $2n + g - 1$ places of degree one P_i which are not in the support of D such that the multi-evaluation map T defined by

$$\begin{aligned} T:\ \mathcal{L}(2D) &\to \left(\mathbb{F}_q\right)^{2n+g-1} \\ f &\mapsto \left(f(P_1), \ldots, f\left(P_{2n+g-1}\right)\right) \end{aligned}$$

is an isomorphism.

2.1.1. Strategy of implementation

The construction of the algorithm is based on the choice of the place Q of degree n, the effective divisor D of degree $n + g - 1$, the bases of spaces $\mathcal{L}(D)$ and $\mathcal{L}(2D)$, and the basis of the residue class field F_Q of the place Q. The place Q of degree n is lying above a normal primitive polynomial in $\mathbb{F}_q[X]$, which is totally decomposed in the algebraic function field $F/\mathbb{F}_q$.

As the residue class field F_Q of the place Q is isomorphic to the finite field $\mathbb{F}_{q^n}$, we identify $\mathbb{F}_{q^n}$ to F_Q. Indeed, $\deg(D) = n + g - 1$, $dim(D - Q) = 0$ yet $\mathcal{L}(D - Q) = Ker(E)$. In particular, we choose for basis of $\mathcal{L}(D)$, the reciprocal image $\mathcal{B}_D$ of the basis $\mathcal{B}_Q = (\phi_1, \ldots, \phi_n)$ of F_Q by the evaluation map E, namely $\mathcal{B}_D = (E^{-1}(\phi_1), \ldots, E^{-1}(\phi_n))$.

Note that as the divisor D is an effective divisor, we have $\mathcal{L}(D) \subset \mathcal{L}(2D)$. Let P be the map from $\mathcal{L}(2D)$ to $\mathcal{L}(2D)$ defined in the following way: if $f \in \mathcal{L}(2D)$ then $f(Q)$ is in the residue field F_Q of the place Q; define $P(f) = J \circ E^{-1}\left(f(Q)\right)$, where J is the injection map from $\mathcal{L}(D)$ into $\mathcal{L}(2D)$. Then P is a linear map from $\mathcal{L}(2D)$ into $\mathcal{L}(2D)$ whose image is $\mathcal{L}(D)$. More precisely, P is a projection from $\mathcal{L}(2D)$ onto $\mathcal{L}(D)$. Let $\mathcal{M}$ be the kernel of P. Then $\mathcal{L}(2D) = \mathcal{L}(D) \oplus \mathcal{M}$.

2.1.2. Product of two elements in $\mathbb{F}_{q^n}$

Let $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ be two elements of $\mathbb{F}_{q^n}$ given by their components over $\mathbb{F}_q$ relative to the chosen basis $\mathcal{B}_Q$. According to the previous notation, we can consider that x and y are identified to the following elements of $\mathcal{L}(D)$:

$$f_x = \sum_{i=1}^{n} x_i f_i \quad \text{and} \quad f_y = \sum_{i=1}^{n} y_i f_i.$$

We will consider that x and y are respectively the elements f_x and f_y of $\mathcal{L}(2D)$ where the $n + g - 1$ last components are 0. Now it is clear that knowing x or f_x by their coordinates is the same thing.

Denote the Hadamard product in $\left(\mathbb{F}_q\right)^{2n+g-1}$ by:

$$(u_1, \ldots, u_{2n+g-1}) \odot (v_1, \ldots, v_{2n+g-1}) = (u_1 v_1, \ldots, u_{2n+g-1} v_{2n+g-1}).$$

Theorem 2.5. *The product of x by y is such that*

$$f_{xy} = P\left(T^{-1}\left(T(f_x) \odot T(f_y)\right)\right).$$

Please cite this article in press as: K. Atighehchi et al., Effective arithmetic in finite fields based on Chudnovsky's multiplication algorithm, C. R. Acad. Sci. Paris, Ser. I (2016), http://dx.doi.org/10.1016/j.crma.2015.12.001

We can now present the setup algorithm (which is only done once) and the multiplication algorithm.

Algorithm 1 Setup algorithm.

INPUT: $F/\mathbb{F}_q$, $Q, D, P_1, \ldots, P_{2n+g-1}$.
OUTPUT: T and T^{-1}.
(i) The elements x of the field $\mathbb{F}_{q^n}$ are known by their components relatively to a fixed basis: $x = (x_1, \ldots, x_n)$ (where $x_i \in \mathbb{F}_q$).
(ii) The function field $F/\mathbb{F}_q$, the place Q, the divisor D and $P_1, \ldots, P_{2n+g-1}$ are as in Theorem 2.4.
(iii) Construct a basis $(f_1, \ldots, f_n, f_{n+1}, \ldots, f_{2n+g-1})$ of $\mathcal{L}(2D)$ where $(f_1, \ldots, f_n)$ is the basis of $\mathcal{L}(D)$ defined in section 2.1.1 and $(f_{n+1}, \ldots, f_{2n+g-1})$ a basis of $\mathcal{M}$.
(iv) Any element $x = (x_1, \ldots, x_n)$ in $\mathbb{F}_{q^n}$ is identified to the element $f_x = \sum_{i=1}^{n} x_i f_i$ of $\mathcal{L}(D)$.
(v) Compute the matrices T and T^{-1}.

Algorithm 2 Multiplication algorithm.

INPUT: $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$.
OUTPUT: xy.
(i) Compute

$$z = \begin{pmatrix} z_1 \\ \vdots \\ z_n \\ z_{n+1} \\ \vdots \\ z_{2n+g-1} \end{pmatrix} = T \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ 0 \\ \vdots \\ 0 \end{pmatrix} \text{ and } t = \begin{pmatrix} t_1 \\ \vdots \\ t_n \\ t_{n+1} \\ \vdots \\ t_{2n+g-1} \end{pmatrix} = T \begin{pmatrix} y_1 \\ \vdots \\ y_n \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

(ii) Compute $u = (u_1, \ldots, u_{2n+g-1})$ where $u = z \odot t$.
(iii) Compute $w = (w_1, \ldots, w_{2n+g-1}) = T^{-1}(u)$.
(iv) Return $(xy = (w_1, \ldots, w_n))$ (in the previous step just the n first components have to be computed).

Our exponentiation is based on our multiplication and a modified algorithm from von Zur Gathen [7].

References

[1] K. Atighehchi, S. Ballet, A. Bonnecaze, R. Rolland, Arithmetic in finite fields based on Chudnovsky multiplication algorithm, preliminary version arXiv: 1510.00090, submitted for publication.
[2] S. Ballet, Curves with many points and multiplication complexity in any extension of $\mathbb{F}_q$, Finite Fields Appl. 5 (1999) 364–377.
[3] D. Chudnovsky, G. Chudnovsky, Algebraic complexities and algebraic curves over finite fields, J. Complex. 4 (1988) 285–316.
[4] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progressions, J. Symb. Comput. 9 (3) (1990) 251–280.
[5] J.-M. Couveignes, R. Lercier, Elliptic periods for finite fields, Finite Fields Appl. 15 (1) (2009) 1–22.
[6] S. Gao, J. von zur Gathen, D. Panario, V. Shoup, Algorithms for exponentiation in finite fields, J. Symb. Comput. 29 (6) (2000) 879–889.
[7] J. von zur Gathen, Efficient and optimal exponentiation in finite fields, Comput. Complex. 1 (1991) 360–394.

Please cite this article in press as: K. Atighehchi et al., Effective arithmetic in finite fields based on Chudnovsky's multiplication algorithm, C. R. Acad. Sci. Paris, Ser. I (2016), http://dx.doi.org/10.1016/j.crma.2015.12.001