A CRT-Based Montgomery Multiplication for Finite Fields of Small Characteristic

Jean-Claude Bajard, Laurent Imbert, Graham A. Jullien, Hugh C. Williams

A CRT-Based Montgomery Multiplication for Finite Fields of Small Characteristic

Jean-Claude Bajard*, Laurent Imbert*,†, Graham A. Jullien† and Hugh C. Williams†
*LIRMM, CNRS UMR 5506, 161 rue Ada, 34392 Montpellier cedex 5, France
†University of Calgary, 2500 University drive NW, Calgary, AB, T2N 1N4, Canada

***Abstract*— We propose a new CRT-based multiplication algorithm for finite fields $\mathbb{F}_{p^k}$ of small prime characteristic, whose complexity does not depend on a special form of the reduction polynomial. With a complexity of $O(k^{3/2})$ this is the first general subquadratic algorithm for fields of small odd characteristic.**

I. Introduction

Finite field arithmetic is an important prerequisite for elliptic curve cryptography (ECC) [1], [2]. ECC has recently received a lot of attention, because of its smaller key-size (the security provided by a 160-bit key is equivalent to a 80-bit symmetric-key for block ciphers or a 1024-bit RSA modulus) and improved theoretical robustness (there is no known subexponential algorithm to solve the ECDLP which is the foundation of ECC). As witness of its commercial acceptance, is its recent inclusion in various standards (see [3], [4]). These standards recommend a small number of secure curves defined over large prime fields $\mathbb{F}_p$ and binary fields $\mathbb{F}_{2^m}$. As a consequence, the use of other finite fields of potential interest has received limited attention.

In this paper we propose a modified Montgomery multiplication algorithm for finite fields of small prime characteristic, i.e. $\mathbb{F}_{p^k} \simeq \mathbb{F}_p[X]/(N)$, where N is a monic irreducible polynomial of degree k, called the reduction polynomial. Is is known that the elements of $\mathbb{F}_{p^k}$ can be modeled as polynomials of degree at most $k-1$, with coefficients in $\mathbb{F}_p$. (It is easy to verify that there are exactly p^k such polynomials.) The arithmetic operations (addition, multiplication) are carried out using polynomial arithmetic modulo N. In this work, we represent the operands in a so-called *Chinese Remainder Representation* (CR), where polynomials of degree less than k are represented using their remainders modulo m relatively prime binomials of degree d, of the form $X^d + c$, where $c \in \mathbb{F}_p$. Our algorithm requires $p > 2m$ and $md \geq k$. Compared to the classical Montgomery algorithm, our algorithm requires fewer multiplications in $\mathbb{F}_p$, with best results obtained for fields of small characteristic. If m, d are chosen close to $\sqrt{k}$, we obtain a subquadratic complexity of $O(k^{1.5})$, whereas Montgomery's algorithm is in $O(k^2)$.

II. Montgomery multiplication in $\mathbb{F}_{p^k}$

In this section, we briefly recall the Montgomery's modular multiplication algorithm for integers, and its straightforward extension to finite fields. Then, we present a new algorithm which is a generalization of Montgomery multiplication, where the elements of the finite field $\mathbb{F}_{p^k}$ are represented in polynomial residue arithmetic, i.e. according to their remainders modulo a set of well chosen, relatively prime polynomials.

Let us start with Montgomery's multiplication algorithm for integers [5]. Instead of computing $ab \bmod n$, Montgomery's algorithm returns[1] $s = abr^{-1} \bmod n$, where r is such that $\gcd(r, n) = 1$. In practice n is always an odd number, and r is chosen as a power of 2 to reduce multiplication and division by r to simple shifts. The number r is often referred to as the Montgomery factor. The computation is accomplished in two steps: first

[1]We use the notation $x = y \bmod n$ to denote the remainder $x < n$ in the division of y by n; and the notation $x \equiv y \pmod n$ to express the fact that x and y have the same remainder modulo n.

compute $q = -abn^{-1} \bmod r$, then $ab + qn$ is a multiple of r; hence, an exact division by r, which is equivalent to some right shifts, gives the result. In order to get an actual product, $ab \bmod n$, different methods are possible. If s is the output of the Montgomery product of a and b modulo n, i.e., $s = MM(a, b, n)\,(= abr^{-1} \bmod n)$, then, it is easy to see that $MM(s, r^2 \bmod n, n) = a\,b \bmod n$. Also, $MM(a, br \bmod n, n) = MM(ar \bmod n, b, n) = a\,b \bmod n$. When several multiplications need to be performed modulo the same n, as in an exponentiation, the inputs are first modified using the transformation, $a \rightarrow ar \bmod n$, which has the advantage of being stable over Montgomery multiplication (indeed, we have $MM(ar \bmod n, br \bmod n, n) = a\,b\,r \bmod n$). These conversions can be done at the extra cost of only two Montgomery multiplications (with $r^2 \bmod n$ and 1 as inputs respectively), which is negligible compared to the cost of the exponentiation. See [6] for more details.

Montgomery's multiplication of large integers has been generalized to binary fields $\mathbb{F}_{2^k}$ by Ç. K. Koç and T. Acar [7]. Their solution is a direct adaptation of the original Montgomery algorithm, where the polynomial X^k plays the role of the Montgomery factor r. Given $A, B \in \mathbb{F}_{2^k}$, the algorithm computes $ABX^{-k} \bmod N$, where N is the reduction polynomial of degree k in $\mathbb{F}_2[X]$. We remark that Koç and Acar's algorithm easily extends to any extension fields $\mathbb{F}_{p^k}$. As in [7], we represent $\mathbb{F}_{p^k}$ with respect to a monic irreducible polynomial N of degree k and we consider its elements as polynomials of degree at most $k-1$ in $\mathbb{F}_p[X]$. Let us define $\Psi = X^k$ such that $\gcd(\Psi, N) = 1$. Then, given $A, B \in \mathbb{F}_p[X]/(N)$, Algorithm 1 below, returns $AB\Psi^{-1} \bmod N$.

Algorithm 1 Montgomery Multiplication over $\mathbb{F}_{p^k}$

Input: Two polynomials $A, B \in \mathbb{F}_p[X]$, with $\deg A, \deg B \leq k-1$; a monic irreducible polynomial $N \in \mathbb{F}_p[X]$, with $\deg N = k$; a polynomial $\Psi = X^k$

Output: $A\,B\,\Psi^{-1} \bmod N$

1: $Q = -A \times B \times N^{-1} \bmod \Psi$

2: $R = (A \times B + Q \times N)\,/\Psi$

In this case, selecting $\Psi = X^k$ seems to be the perfect choice, since the reduction modulo X^k (in Step 1) and the division by X^k (in Step 2) can be easily implemented. Indeed, given two polynomials $U, V \in \mathbb{F}_p[X]$, with $\deg U, \deg V < k$, we compute $(U \times V) \bmod X^k$ by ignoring the coefficients of $U \times V$ of order larger than $k-1$. Similarly, $(U \times V)/X^k$ is given by the coefficients of $(U \times V)$ of order greater than or equal to k.

The complexity, in terms of the number of arithmetic operations over $\mathbb{F}_p$, can be easily determined (see [8] for details). The computation of Q in Step 1 requires $k(k+1)$ multiplications, and $k(k-1)$ additions; whereas R, in Step 2, is computed in $k(k-1)$ multiplications, and $\frac{(k-1)(k-2)}{2} + \frac{k(k-1)}{2} + (k-1)$ additions. If M and A denote the costs of one multiplication and one addition in $\mathbb{F}_p$ respectively, the total cost for Algorithm 1 is thus

$$2k^2\,M + (2k^2 - 2k)\,A. \tag{1}$$

For most applications (especially for ECC) the finite field is fixed and we can reasonably assume that the reduction polynomial N and its inverse modulo X^k have been precomputed. In this case, the multiplications by the N^{-1} and N can be simplified using optimized algorithms for multiplication by a constant and by constant vectors, as in [9]. The global cost of Algorithm 1 becomes

$$k^2\,M + k^2\,CM + (2k^2 - 2k)\,A, \tag{2}$$

where CM denotes the cost of one multiplication by a constant in $\mathbb{F}_p$.

III. Modified Montgomery multiplication in $\mathbb{F}_{p^k}$

In this section we first modify Algorithm 1 by allowing the polynomial Ψ to be any polynomial of degree k satisfying the condition $\gcd(\Psi, N) = 1$ and by replacing the division by Ψ in Step 2 by a multiplication by Ψ^{-1} modulo another given polynomial Ψ'. Then we analyze a special case, where Ψ, Ψ' are the products of relatively prime polynomials of small degree; and Ψ, Ψ' are relatively prime. Algorithm 2 below computes $AB\Psi^{-1} \bmod N$ for any relatively prime polynomials Ψ, Ψ', both of degree k with $\gcd(\Psi, N) = 1$.

Algorithm 2 Modified Montgomery Multiplication over $\mathbb{F}_{p^k}$

Input: Two polynomials $A, B \in \mathbb{F}_p[X]$, with $\deg A, \deg B \leq k-1$; a monic irreducible polynomial $N \in \mathbb{F}_p[X]$, with $\deg N = k$; two polynomials Ψ, Ψ', with $\deg \Psi = \deg \Psi' \geq k$, and $\gcd(\Psi, \Psi') = \gcd(\Psi, N) = 1$

Output: $AB\Psi^{-1} \bmod N$

1: $Q = -A \times B \times N^{-1} \bmod \Psi$

2: $R = (A \times B + Q \times N) \times \Psi^{-1} \bmod \Psi'$

Lemma 1: Algorithm 2 is correct and returns $AB\Psi^{-1} \bmod N$.

Proof: In Step 1, we compute Q such that $(AB + QN)$ is a multiple of Ψ. Indeed, we have $AB + QN \equiv AB - ABN^{-1}N \equiv 0 \pmod{\Psi}$ and $\deg AB + QN = \deg QN \leq 2k - 1$. This implies that there exists a polynomial f such that $AB + QN = f\Psi$, with $\deg f \leq k-1$. Now, in step 2, we compute R modulo Ψ'. We have $(AB+QN)\Psi^{-1} \equiv (f\Psi)\Psi^{-1} \equiv f \pmod{\Psi'}$. Since $\deg \Psi' \geq k > \deg f$, we have $(AB+QN)\Psi^{-1} \bmod \Psi' = f = R$. In general, for any polynomial h with $\deg h \geq k$, and $\gcd(h, \Psi) = 1$, we have $(AB+QN)\Psi^{-1} \bmod h = f$. In particular, for $h = N$ we have $(AB + QN)\Psi^{-1} \equiv AB\Psi^{-1} \pmod{N}$ which concludes the proof. □

Of course, the generalization proposed in Algorithm 2 is interesting, only if we can define polynomials Ψ, Ψ' such that the arithmetic operations modulo Ψ and Ψ' are easy to implement.

IV. New algorithm

In this section, we define Ψ (resp. Ψ') to be the product of some relatively prime binomials of the form $X^d + c$, where $c \in \mathbb{F}_p$, and we use the Chinese Remainder Theorem (CRT) in order to represent the elements (polynomials) of $\mathbb{F}_p[X]$ of degree less than k by their remainders modulo sufficiently many of these binomials. The advantage is that we distribute the costly arithmetic modulo Ψ (resp. modulo Ψ') into several independent arithmetic units, each performing its arithmetic modulo a very simple polynomial.

Suppose that $\Psi = \prod_{i=1}^{m} \psi_i$, where $\psi_i = X^d + c_i$, with $c_i \in \mathbb{F}_p$, and $c_i \neq c_j$ for $i \neq j$. [2] Then, for any arbitrary $U \in \mathbb{F}_p[X]$, we define $u_i = U \bmod (X^d + c_i)$. The following ring isomorphism given by the Chinese Remainder Theorem

$$\begin{array}{ccc} \mathbb{F}_p[X]/(\Psi) & \rightarrow & \mathbb{F}_p[X]/(\psi_1) \times \cdots \times \mathbb{F}_p[X]/(\psi_m) \\ U & \mapsto & (u_1, \ldots, u_m)\,, \end{array} \tag{3}$$

tells us that if $\deg U < \deg \Psi = dm$, then U is uniquely defined by its remainders $(u_1, \ldots, u_m)$ modulo $\psi_1, \ldots, \psi_m$.

Definition 1: Let $U \in \mathbb{F}_p[X]$ with $\deg U < k$, and let $\Psi = \prod_{i=1}^{m}(X^d + c_i)$, with $c_i \in \mathbb{F}_p$ and $c_i \neq c_j$ for $i \neq j$. We define the *Chinese Remainder (CR) representation* of U modulo Ψ as

$$CR_{\Psi}(U) = (u_1, \ldots, u_m), \tag{4}$$

where $u_i = U \bmod (X^d + c_i)$ is a polynomial in $\mathbb{F}_p[X]$ of degree at most $d-1$, for $i = 1, \ldots, m$.

One advantage of the CR representation, is that the arithmetic modulo Ψ is carried out implicitly by performing the arithmetic modulo each ψ_i independently. In Algorithm 2 we need another polynomial Ψ'. We define $\Psi' = \prod_{i=1}^{m}(X^d + c'_i)$, where $c'_i \neq c'_j$ for $i \neq j$ and $c_i \neq c'_j$ for $i, j = 1, \ldots, m$. This simply means that the c_i's and c'_i's are all distinct. With Ψ and Ψ' defined as above, we have $\gcd(\Psi, \Psi') = 1$, and because N is irreducible in $\mathbb{F}_p$, we also have $\gcd(\Psi, N) = 1$.

We assume that the input polynomials A, B are given (or converted into) the CR representation modulo both Ψ and Ψ'. In Step 1, we also need $CR_{\Psi}(N^{-1})$, which can be precomputed. We note that since $\gcd(\Psi, N) = 1$, then N^{-1} always exists modulo Ψ. For the operations in Step 2, we also need $CR_{\Psi'}(\Psi^{-1})$ which also always exists, and $CR_{\Psi'}(N)$. Both can be precomputed. We remark further that $(\Psi^{-1} \bmod \psi_i) \in \mathbb{F}_p$, for $i = 1, \ldots, m$. The only problem to solve is the conversion of Q from its residue representation modulo Ψ to its residue representation modulo Ψ'. Similarly, if we wish to reuse the output of the multiplication, as in

[2] Note that with Ψ defined as above, we clearly need $m < p$. Moreover, since we shall also define $\Psi' = \prod_{i=1}^{m}(X^d + c'_i)$ such that $\gcd(\Psi, \Psi') = 1$, this condition will become $2m < p$.

the performance of an exponentiation, we need to convert R back from $CR_{\Psi'}$ to CR_{Ψ}.

A straightforward approach is to use the constructive proof of the Chinese remainder theorem to convert Q from its CR representation to its classical polynomial (coefficient based) representation, and then reduce it modulo each ψ'_i to get $CR_{\Psi'}(Q)$. Another solution, however, which is much more efficient in this case, is to use Newton's interpolation algorithm.[3]

Assume $CR_{\Psi}(Q) = (q_1, \ldots, q_m)$. We want to compute $CR_{\Psi'}(Q) = (q'_1, \ldots, q'_m)$. We must first evaluate the intermediate values $(\zeta_1, \ldots, \zeta_m)$ – often referred to as the mixed-radix representation in the integer case – where the ζ_i's are polynomials of degree less than d. The vector $(\zeta_1, \ldots, \zeta_m)$ is obtained by performing the following computations:

$$\begin{cases} \zeta_1 = q_1 \\ \zeta_2 = (q_2 - \zeta_1)\,\psi_1^{-1} \bmod \psi_2 \\ \zeta_3 = \big((q_3 - \zeta_1)\,\psi_1^{-1} - \zeta_2\big)\,\psi_2^{-1} \bmod \psi_3 \\ \quad \vdots \\ \zeta_m = \big(\ldots\big((q_m - \zeta_1)\,\psi_1^{-1} - \zeta_2\big)\,\psi_2^{-1} - \cdots \\ \qquad - \zeta_{m-1}\big)\psi_{m-1}^{-1} \bmod \psi_m. \end{cases} \tag{5}$$

We then evaluate the polynomials q'_i's using Horner's rule as

$$q'_i = (\ldots((\zeta_m\psi_{m-1} + \zeta_{m-1})\psi_{m-2} + \cdots + \zeta_3)\psi_2 + \zeta_2)\psi_1 + \zeta_1 \bmod \psi'_i. \tag{6}$$

Concerning (5), we remark that the main operation is a polynomial multiplication of the form $U \times \psi_i^{-1} \bmod \psi_j$ (e.g. for ζ_2, we have $U = q_2 - \zeta_1$). In (6), the main operation is also a polynomial multiplication of the form $V \times \psi_i \bmod \psi'_j$. With ψ_i, ψ'_j defined as above, i.e., $\psi_i = X^d + c_i$ and $\psi'_j = X^d + c'_j$, we have

$$\psi_i \bmod \psi_j = c_i - c_j \bmod p \quad \in \mathbb{F}_p. \tag{7}$$

And thus,

$$\psi_i^{-1} \bmod \psi_j = (c_i - c_j)^{-1} \bmod p \quad \in \mathbb{F}_p. \tag{8}$$

[3]In fact, it is useful to think of the CRT as a special case of interpolation.

As a consequence, (5) and (6) can be evaluated without any polynomial multiplications! The only required operations are the addition of two polynomials of degree less than d and the multiplication of a polynomial of degree less than d by an element of $\mathbb{F}_p$. Both are very easy operations that can be fully parallelized. In the complexity analysis of our multiplication algorithm, we will only consider the number of real polynomial multiplications and the number of multiplications by integer constants. For the computation of the ζ_i's, we have to evaluate $m(m-1)/2$ expressions of the form $U \times \psi_i^{-1} \bmod \psi_j = U \times (c_i - c_j)^{-1}$, since $\deg U < d$. Thus, it requires $dm(m-1)/2\ CM$, where CM denotes the cost of one multiplication by an integer constant in $\mathbb{F}_p$. For (6), we have to compute $(m-1)$ expressions of the form $V \times \psi_i \bmod \psi_j = V \times (c_i - c_j)$ for each q'_i, where $\deg V < k$. The cost is thus $dm(m-1)\ CM$. The total cost for one Newton's interpolation is thus equal to

$$C_{Newton} = \frac{3}{2}\, d\, m(m-1)\ CM. \tag{9}$$

Let us now analyze the cost of our modified Montgomery algorithm with Ψ, Ψ' defined as above. Given $CR_{\Psi}(A)$ and $CR_{\Psi}(B)$, Algorithm 2 first computes $CR_{\Psi}(A \times B) = (t_1, \ldots, t_m)$, where $t_i = (a_i \times b_i) \bmod \psi_i$. The cost is m polynomial multiplications modulo a polynomial of the form $X^d + c$. In order to provide a more accurate complexity measure, let us make precise the cost of our main operation; i.e., the product

$$(a_i \times b_i) \bmod (X^d + c_i), \tag{10}$$

where $\deg a_i, \deg b_i \leq d - 1$. Assuming we first compute the product $s_i = (a_i \times b_i)$ using the basic high-school algorithm, we need d^2 multiplications in $\mathbb{F}_p$. (Note that when d is large, it might be more interesting to use a more efficient polynomial multiplication algorithm, such as Karatsuba-like methods [10].) The result s_i is a polynomial of degree at most $2d-2$. The reduction part is performed using the congruence $X^d \equiv -c_i \bmod \psi_i$. We have

$$s_i = (s_i \bmod X^d) - c_i\,(s_i - s_i \bmod X^d)/X^d. \tag{11}$$

The reduction modulo X^d and the division by X^d are easy operations, performed at no cost. They simply consists of ignoring the terms of s_i of order larger than $d-1$ (for the reduction modulo X^d), and considering only those terms of order larger than or equal to d for the division. Since $\deg s_i \leq 2d-2$, the multiplication of the higher part of s_i by c_i requires $d-1$ CM. Thus, the cost for (10) is equal to d^2 $M + (d-1)$ CM. In the following analysis, we will also encounter some polynomial multiplications modulo $X^d + c$, where one of the operands is a constant polynomial. In these cases, the cost is d^2 $CM + (d-1)$ CM.

After $CR_\Psi(A \times B)$ is computed, we evaluate $CR_\Psi(Q) = (q_1, \ldots, q_m)$, where $q_i = (-t_i \times \tilde{n}_i) \bmod \psi_i$, and $(\tilde{n}_1, \ldots, \tilde{n}_m) = CR_\Psi(N^{-1})$ is the CR representation of N^{-1} modulo Ψ. Because one of the operands ($CR_\Psi(N^{-1})$) is a constant polynomial, the cost is md^2 $CM + m(d-1)$ CM. The total cost for Step 1 is therefore

$$C_1 = md^2\ M + \left(md^2 + 2m(d-1)\right)\ CM. \quad (12)$$

After step 1, Q is converted to $CR_{\Psi'}(Q) = (q'_1, \ldots, q'_m)$ using Newton's interpolation algorithm whose cost is given by (9).

For Step 2, we require m polynomial multiplications to compute $CR_{\Psi'}(A \times B)$, plus m constant polynomial multiplications ($CR_{\Psi'}(N)$). Instead of performing the reduction twice, i.e., after each multiplication, we only need to reduce their sum $AB+QN$. We terminate Step 2 with the multiplication by $\Psi^{-1} \bmod \Psi'$, which requires only md CM operations since $\Psi^{-1} \bmod \psi'_i$ belongs to $\mathbb{F}_p$, for $i = 1, \ldots, m$. Indeed, we have

$$\begin{aligned}
\Psi^{-1} \bmod \psi'_i &= \left(\prod_{j=1}^{m} \psi_j\right)^{-1} \bmod \psi'_i \\
&= \left(\prod_{j=1}^{m} (\psi_j \bmod \psi'_i)\right)^{-1} \bmod \psi'_i \\
&= \left(\prod_{j=1}^{m} (c_j - c'_i)\right)^{-1} \bmod p \quad \in \mathbb{F}_p.
\end{aligned}$$

The cost of Step 2 is thus

$$C_2 = md^2\ M + \left(md^2 + m(d-1) + md\right)\ CM. \quad (13)$$

The last step of our multiplication algorithm consists of the conversion of R from its CR representation modulo Ψ' to its CR representation modulo Ψ using Newton's interpolation algorithm.

The total cost of Algorithm 2 is thus equal to

$$\begin{aligned}
C_{m,d} &= 2md^2\ M \\
&\quad + \left(2md^2 + 3dm^2 + md - 3m\right)\ CM. \quad (14)
\end{aligned}$$

In Table I below, we illustrate the efficiency of our algorithm for fields of small characteristic. We give examples of possible decompositions m, d, which lead to fewer multiplications in $\mathbb{F}_p$ than the $2k^2$ required by Algorithm 1. For each example, we give the parameters m and d such that $md \geq k$, and there exists a prime p yielding a field $\mathbb{F}_{p^k}$ of cryptographic interest (for elliptic curve applications [11]). For security reasons, we only consider extensions of prime degree k. (See [12] for a recent attack when $k \equiv 0 \bmod 3$ or 4.) We use l to denote the corresponding key-size $l = \lfloor \log_2(p^k) \rfloor$ bits. Note that p and m satisfy the condition $2m < p$. For simplicity in the comparisons, we further assume that $CM = M$.

From Table I we remark that it is always possible to define convenient pairs m, d for $p \geq 5$. However, in the case $p = 5$, we cannot consider more than two binomials ($m = 2$ is the only option), and Algorithm 1 always requires fewer multiplications in the general case.[4]

For small p, it is reasonable to assume that $CM = M$, since the arithmetic modulo p can be implemented very efficiently; e.g. for hardware implementations, using a small lookup table. Also, when m is small, one can optimize the choice of the c_i's to save some multiplications. For large p, however, one can reasonably consider that $CM < M$. Efficient algorithms for multiplication by an integer constant have been investigated [9] and can be used in this context to further improvethe efficiency of our technique.

[4]For $m = 2$ and $md = k + 1$, we have $C_{m,n} = 2k^2 + 11k + 3$.

TABLE I

NUMBER OF MULTIPLICATIONS OBTAINED FOR VARIOUS PARAMETERS m, d, WITH $md \geq k$ AND $2m < p$, AND WHERE p AND k ARE PRIMES SUCH THAT $\mathbb{F}_{p^k}$ IS A FIELD OF CRYPTOGRAPHIC INTEREST; I.E. OF ORDER $\geq 2^{160}$

p	k	l	m	d	$2k^2$	$C_{m,d}$
5	71	164	2	36	10082	10866
7	59	165	3	20	6962	5391
11	47	162	5	10	4418	2785
			4	12		2914
13	47	173	6	8	4418	2430
			5	10		2785
			4	12		2916
17	41	167	8	6	3362	2328
			7	6		1911
			6	7		1956
19	41	174	9	5	3362	2133
			8	6		2328
			7	6		1911
			6	7		1956
23	37	167	8	5	2738	1776
			7	6		1911
			6	7		1956
			5	8		1905
127	23	160	12	2	1058	1044
			8	3		864
			6	4		822
			5	5		885
			4	6		876

From Table I we notice that we get the best results when m and d are close. Indeed, if we assume that $m, d \approx k^{1/2}$, then under the assumption that $CM = M$, we obtain a subquadratic complexity:

$$C_k = 7k^{3/2} + k - 3k^{1/2} = O(k^{3/2}). \qquad (15)$$

V. EXAMPLE

In the following example, we assume $p = 7$, $k = 13$ ($7^{13} \approx 2^{36}$). Let $A = 5X^{12} + 2X^{10} + 3X^8 + 2X^4 + 4X$, and $B = X^9 + X^8 + 2X^5 + 4X^3 + 2X^2 + 1$, be two polynomials in the field $\mathbb{F}_7[X]/(N)$, where $N = X^{13} + 2X^4 + 4X + 2$ is irreducible.

We consider the parameters $m = 3$, $d = 5$ and the polynomials $\Psi = (X^3 + 1)(X^3 + 2)(X^3 + 3)$, and $\Psi' = (X^3 + 4)(X^3 + 5)(X^3 + 6)$. To simplify the notations, we represent polynomials (of degree at most 4) by their coefficients only (in $\mathbb{F}_7$), where the right-most digit correspond to the coefficient of degree 0. The precomputed values, expressed in CR representations are

$$CR_\Psi(-N^{-1}) = (15016, 65413, 64652)$$
$$CR_{\Psi'}(N) = (22402, 24042, 21042)$$
$$CR_{\Psi'}(\Psi^{-1}) = (00001, 00002, 00005).$$

Note that $\Psi^{-1} \bmod \psi'_i \in \mathbb{F}_p$ for $i = 1, \ldots, 3$. We express A, B in CR representation for both Ψ and Ψ'. We have

$$CR_\Psi(A) = (24542, 21641, 25344)$$
$$CR_{\Psi'}(A) = (22344, 26641, 23542)$$
$$CR_\Psi(B) = (63206, 52204, 41202)$$
$$CR_{\Psi'}(B) = (30200, 26205, 15203).$$

And we compute

$$CR_\Psi(Q) = (32505, 02004, 36666)$$
$$CR_{\Psi'}(Q) = (50244, 65215, 60642)$$
$$CR_\Psi(R) = (43601, 15003, 50042)$$
$$CR_{\Psi'}(R) = (22655, 64435, 36152).$$

It is easy to check that the result is the CR representation of $A \times B \times \Psi^{-1} \bmod N = 3X^{12} + 2X^{11} + 2X^{10} + 3X^9 + 5X^8 + X^7 + 6X^6 + 4X^5 + X^3 + 4X^2 + 4X + 3$, modulo Ψ and Ψ'.

VI. CONCLUSIONS

In this paper, we have proposed a modified Montgomery multiplication algorithm for finite fields, where the operands are represented by their remainders modulo a set of relatively prime binomials of the form $X^d + c$, with $c \in \mathbb{F}_p$. Our algorithm takes advantage of the fact that $X^d + c_1 \bmod (X^d + c_2) = c_1 - c_2 \in \mathbb{F}_p$, and it requires fewer multiplications in the ground field that its classical counterpart. Indeed, if $m, d \approx \sqrt{k}$, we can obtain a subquadratic complexity of $O(k^{3/2})$ for Montgomery multiplication.

ACKNOWLEDGMENTS

This work was done during L. Imbert leave of absence at the university of Calgary, with the *Centre for Information Security and Cryptography (CISaC)* and the *Advanced Technology Information Processing Systems (ATIPS)* Laboratory. It was partly supported by NSERC Canada, under the strategic grant number 73–2048, *Novel implementation of cryptographic algorithms on custom hardware platforms*; and by the French ministry of education and research under the grant *ACI Sécurité Informatique 2003, Opérateurs Cryptographiques et Arithmétique Matérielle (OCAM)*.

REFERENCES

[1] V. S. Miller, "Uses of elliptic curves in cryptography," in *Advances in Cryptology – CRYPTO '85*, ser. LNCS, H. C. Williams, Ed., vol. 218. Springer-Verlag, 1986, pp. 417–428.

[2] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, January 1987.

[3] National Institute of Standards and Technology, *FIPS PUB 186-2: Digital Signature Standard (DSS)*. National Institute of Standards and Technology, Jan. 2000.

[4] IEEE, *IEEE 1363-2000 Standard Specifications for Public-Key Cryptography*, 2000.

[5] P. L. Montgomery, "Modular multiplication without trial division," *Mathematics of Computation*, vol. 44, no. 170, pp. 519–521, Apr. 1985.

[6] A. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC Press, 1997.

[7] Ç. K. Koç and T. Acar, "Montgomery multiplication in $GF(2^k)$," *Designs, Codes and Cryptography*, vol. 14, no. 1, pp. 57–69, April 1998.

[8] J.-C. Bajard, L. Imbert, and C. Nègre, "Arithmetic operations in finite fields of medium prime characteristic for elliptic curve cryptography," LIRMM – CNRS UMR 5506, 161 rue Ada, 34392 Montpellier cedex 5, France, Research Report 05028, Mar. 2005, available electronically at http://www.lirmm.fr/~imbert.

[9] V. Lefèvre, "Multiplication by an integer constant," INRIA, Research Report 4192, May 2001.

[10] P. L. Montgomery, "Five, six, and seven-term karatsuba-like formulae," *IEEE Transactions on Computers*, vol. 54, no. 3, pp. 362–369, Mar. 2005.

[11] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. Springer-Verlag, 2004.

[12] P. Gaudry, "Index calculus for abelian varieties and the elliptic curve discrete logarithm problem," Oct. 2004, preprint.