Fast Multivariate Power Series Multiplication in Characteristic Zero

Grégoire Lecerf and Éric Schost

Laboratoire GAGE, École polytechnique 91128 *Palaiseau, France*
E-mail: lecerf,schost@gage.polytechnique.fr

Abstract

Let k be a field of characteristic zero. We present a fast algorithm for multiplying multivariate power series over k truncated in total degree. Up to logarithmic factors, its complexity is optimal, i.e. *linear* in the number of coefficients of the series.

1. Introduction

Let k be a field of characteristic zero. We denote by S the multivariate power series ring in n variables $k[[x_1, \ldots, x_n]]$ and by $\mathfrak{m}$ its maximal ideal $(x_1, \ldots, x_n)$. For any positive integer d we write $\deg(\mathfrak{m}^{d+1})$ for the *degree* of the ideal $\mathfrak{m}^{d+1}$, that is the number of monomials in S which are not in $\mathfrak{m}^{d+1}$. It is well-known that

$$\deg(\mathfrak{m}^{d+1}) = \binom{d+n}{n}.$$

We view a power series f in S at precision $\mathfrak{m}^{d+1}$ as a vector in the k-algebra $S/\mathfrak{m}^{d+1}$; this algebra has dimension $\deg(\mathfrak{m}^{d+1})$.

In this article we give an asymptotically fast algorithm for multiplying two power series at precision $\mathfrak{m}^{d+1}$: the cost of one multiplication is linear in $\deg(\mathfrak{m}^{d+1})$ up to logarithmic factors. As for many other algorithms dedicated to fast multiplication, our method relies on multipoint evaluation and interpolation: this brings back the problem to univariate power series multiplication in $k[[t]]$ at precision t^{d+1}, for which fast algorithms are known.

Applications. Our interest for power series multiplication originates from the field of polynomial system solving. In the second author's PhD thesis [Schost, 2000], the situation is as follows. We consider some polynomials $f_1, \ldots, f_m$ in $k(x_1, \ldots, x_n)[y_1, \ldots, y_m]$, and want to solve the system $f_1 = \cdots = f_m = 0$. The variables x_i play the role of parameters, and we are looking for formulas expressing the y_i in terms of these parameters. If the system is zero dimensional in the algebraic closure of $k(x_1, \ldots, x_n)$, we proceed the following way: we pick up a point $p_1, \ldots, p_n$ at random in k^n, we substitute the variables x_i by p_i in the system and solve this specialized system. We then lift the dependency of the solutions in the parameters in the formal power series ring $k[[x_1 - p_1, \ldots, x_n - p_n]]$, using a refined version of Newton's iterator, as proposed by Giusti et al. [2001] and Lecerf [2001a,b]. Once we have reached a sufficient precision we recover the solutions thanks to a multivariate version of Padé's approximants. The lifting step is the bottleneck of this method and this is where we really need fast multivariate power series multiplication.

In a similar spirit, multiplication routines modulo $\deg(\mathfrak{m}^{d+1})$ are useful to treat systems of partial differential equations: roughly speaking, once a characteristic set of the system is known, the Taylor series expansions of non-singular solutions can be computed by successive approximations, which require arithmetic operations on power series. This idea is presented for instance by Boulier et al. [1995] and Péladan [1997].

Previous work. To the best of our knowledge, this is the first time that the question of a fast multiplication algorithm is addressed in this context. Apart from the naive algorithm, with complexity quadratic in $\deg(\mathfrak{m}^{d+1})$, the best algorithm known up to now is hinted at by Brent and Kung [1977]: it relies on Kronecker's substitution [1882]. This method requires to compute modulo $(x_1^{d+1}, \ldots, x_n^{d+1})$ instead of $\mathfrak{m}^{d+1}$ and amounts to multiplying two univariate polynomials in degree $(2d)^n$. With respect to the size of the series $\deg(\mathfrak{m}^{d+1})$, the overhead is at least $c2^n n!$, for fixed n, $d \gg n$ and a positive constant c.

Main Result. All along this paper, our model of computation is the *computation tree*, that is the *arithmetic circuit* over k, with operations $(+, \times, \div)$ and branching. In short, we count the operations in k and the tests at unit cost; see Bürgisser et al. (Chapter 4.4) for a precise definition. For a positive integer d, the elements of $S/\mathfrak{m}^{d+1}$ will be represented by the vector of their coefficients in the monomial basis.

With these conventions, our main result is the following:

Theorem 1: *Let d be a positive integer, and let D denote $\deg(\mathfrak{m}^{d+1})$. Let f and g be two elements of $S/\mathfrak{m}^{d+1}$. In terms of operations in k, the product fg has complexity*

$$\mathcal{O}(D \log(D)^3 \log(\log(D))).$$

Our result is closely related to the algorithms for sparse [Ben-Or and Tiwari, 1988] and [Zippel, 1990] or dense [Canny et al., 1989] multivariate polynomial multiplication, which achieve a linear complexity in the number of monomials in the output. All these results rely on a fast multipoint evaluation and interpolation scheme for multivariate polynomials, for a specific choice of sample points, namely powers of prime numbers.

This idea was introduced by Grigoriev and Karpinski [1987] and Tiwari [1987]. We recall this fundamental result in Lemma 1, following the presentation of Canny et al. [1989].

2. Proof of the main result

We first introduce some notation. In the following, C (resp. D) denotes the number of monomials in $n-1$ (resp. n) variables of degree at most d:

$$C := \binom{d+n-1}{n-1}, \qquad D := \binom{d+n}{n}.$$

The log function is the Neperian logarithm ($\log(e) = 1$).

Our proof is divided in three lemmas, the first of which is taken from Canny et al. [1989]. Its result is stated in terms of the function $\mathcal{M}_u(\delta)$, which denotes the complexity of the multiplication of two univariate polynomials of degree δ in $k[t]$. Schönhage and Strassen [1971, 1977] proved that $\mathcal{M}_u(\delta)$ belongs to $\mathcal{O}(\delta \log(\delta) \log(\log(\delta)))$.

LEMMA 1: **Canny et al. [1989]** *Let f be a polynomial in $k[x_2, \ldots, x_n]$ of degree at most d. Let $(p_2, \ldots, p_n)$ be distinct prime numbers. For $i = 0, \ldots, C-1$, we denote by P_i the point $(p_2^i, \ldots, p_n^i)$. There exist computation trees of sizes $\mathcal{O}(\mathcal{M}_u(C)\log(C))$ which perform the following tasks:*

- **Multipoint evaluation:** *compute the values $f(P_0), \ldots, f(P_{C-1})$;*
- **Interpolation:** *recover f from the values $f(P_0), \ldots, f(P_{C-1})$.*

The choice of the points P_i reduces both problems to "transposed" versions of univariate multipoint evaluation and interpolation, for which asymptotically fast solutions exist. These solutions require to invert a Vandermonde matrix built upon all the monomials of degree at most d in $(p_2, \ldots, p_n)$. The key point is that in characteristic zero, all these monomials have pairwise distinct values.

On the basis of this result, the following lemma gives a first upper bound on the complexity of multivariate power series multiplication, stated in terms of the function $\mathcal{M}_u$. In a second stage, we will show that Schönhage-Strassen's multiplication scheme yields the claim of Theorem 1.

LEMMA 2: *Let f and g be two elements of $S/\mathfrak{m}^{d+1}$. In terms of operations in k, the product fg has complexity*

$$\mathcal{O}\Big(d\mathcal{M}_u(C)\log(C) + \mathcal{M}_u(d)C\Big),$$

where C is the number of monomials of degree at most d in $n-1$ variables as defined above.

Proof. Let h be the product fg, and t a new variable. We define F, G, H by

$$F := f(t, x_2t, \ldots, x_nt), \; G := g(t, x_2t, \ldots, x_nt), \; H := h(t, x_2t, \ldots, x_nt).$$

The series F, G and H belong to $k[x_2, \ldots, x_n][[t]]$, and can be written

$$\begin{array}{rcl} F & = & f_0 + f_1t + \ldots + f_dt^d, \\ G & = & g_0 + g_1t + \ldots + g_dt^d, \\ H & = & h_0 + h_1t + \ldots + h_dt^d, \end{array}$$

where for all i, f_i, g_i and h_i belong to $k[x_2, \ldots, x_n]$ and have degrees at most i. The series F, G and H satisfy the equality $H = FG \bmod (t^{d+1})$.

The polynomial h can be recovered from H the following way: $h(x_1t, \ldots, x_nt)$ is obtained by homogenizing each h_i in degree i with respect to the variable x_1, and the evaluation at $t = 1$ yields h.

Consequently, we focus on a fast way to compute H. For any $P = (p_2, \ldots, p_n)$ in k^{n-1}, we write F_P for the series $f_0(P) + f_1(P)t + \ldots + f_d(P)t^d$ in $k[[t]]/(t^{d+1})$, and similarly define G_P and H_P, so that the equality $H_P = F_PG_P \bmod (t^{d+1})$ holds. This leads to the following evaluation-interpolation scheme.

Algorithm. Given f and g in $S/\mathfrak{m}^{d+1}$, to compute $h = fg$ in $S/\mathfrak{m}^{d+1}$.

1. Compute F and G as defined above.
2. Compute (F_{P_i}, G_{P_i}) for C points $(P_0, \ldots, P_{C-1})$, with $P_i \in k^{n-1}$ for all i.
3. Compute the C products $H_{P_i} = F_{P_i}G_{P_i}$ in $k[[t]]/(t^{d+1})$.
4. Compute H by interpolating the polynomials $h_0, \ldots, h_d$.
5. Recover h.

Steps 1 and 5 can be performed by an arithmetic circuit of size linear in C. Following Lemma 1, we now choose $P_i = (p_2^i, \ldots, p_n^i)$, for distinct prime numbers $(p_2, \ldots, p_n)$ and examine the cost of steps 2, 3 and 4 for this specific choice.

- For each j in $0, \ldots, d$, the values

  $$f_j(P_0), \ldots, f_j(P_{C-1}) \text{ and } g_j(P_0), \ldots, g_j(P_{C-1})$$

  can be computed within $\mathcal{O}(\mathcal{M}_u(C)\log(C))$ operations in k using the algorithm for fast multipoint evaluation given in Lemma 1. This yields an overall bound of $\mathcal{O}(d\mathcal{M}_u(C)\log(C))$ operations for step 2.
- For each i in $0, \ldots, C-1$, H_{P_i} is obtained by an univariate series product in $k[[t]]/(t^{d+1})$, which takes $\mathcal{O}(\mathcal{M}_u(d))$ operations; step 3 then requires $\mathcal{O}(\mathcal{M}_u(d)C)$ operations.
- For each j in $0, \ldots, d$, the interpolation of h_j requires $\mathcal{O}(\mathcal{M}_u(C)\log(C))$ operations, using the second result given in Lemma 1. The interpolation of all the h_j then takes $\mathcal{O}(d\mathcal{M}_u(C)\log(C))$ operations. $\square$

The result of Lemma 2 is given in terms of the number of monomials C, whereas our objective is a bound in terms of the quantity D. The last lemma answers this question.

LEMMA 3: *For all $d \geq 0, n \geq 1$, the following inequality holds:*

$$\frac{dC}{D} = \frac{nd}{n+d} \leq \log(D).$$

Proof. The first equality is obvious. Now we rewrite D into $\frac{(n+1)\ldots(n+d)}{d!}$. Then we fix d, and introduce the functions $u : n \mapsto \frac{nd}{n+d}$ and $v : n \mapsto \log \frac{(n+1)\ldots(n+d)}{d!}$. For $n = 1$, the inequality to prove reads $\frac{d}{1+d} \leq \log(1+d)$, which is true. It is immediate to check that $u'(n) \leq v'(n)$ for all n, which implies that $u(n) \leq v(n)$ for all $n \geq 1$. □

We are now ready to prove Theorem 1. We first replace $\mathcal{M}_u(C)$ by the estimate $\mathcal{O}(C \log(C) \log(\log(C)))$ in the complexity of Lemma 2; then, according to the above lemma we bound C by $D\log(D)/d$. This yields a complexity in:

$$\mathcal{O}\Bigg(D\log(D)\Big(\log(D) + \log(\log(D))\Big)^2 \log\Big(\log(D) + \log(\log(D))\Big) \\ + D\log(D)\log(d)\log(\log(d)) \Bigg).$$

As for the second term we use $d \leq D$, therefore:

$$d\mathcal{M}_u(C)\log(C) + \mathcal{M}_u(d)C \in \mathcal{O}(D\log(D)^3 \log(\log(D))).$$

This concludes the proof of Theorem 1.

3. Conclusion

The problem of fast computation with multivariate power series modulo any ideal $\mathfrak{I}$, not necessarily a power of the maximal ideal, is still open. A first question in this direction is the cost of the multiplication modulo the ideal $(x_1^{d+1}, \ldots, x_n^{d+1})$. In this situation, our algorithm requires precision $\mathfrak{m}^{nd+1}$; this yields a complexity in $\mathcal{O}((ed)^n)$, up to logarithmic factors. We do not improve the best complexity result, which is in $\mathcal{O}(\mathcal{M}_u(2d)^n)$ using Kronecker's substitution.

However van der Hoeven [2001] generalized our algorithm to the case when $\mathfrak{I} = (x_1^{d_1} \cdots x_n^{d_n} \mid \alpha_1 d_1 + \cdots + \alpha_n d_n > d)$, where the α_i and d are positive integers.

Moreover our result is stated in terms of arithmetic complexity. It is not immediate to design an efficient implementation of this algorithm. For instance, if we were on $k = \mathbb{Q}$, we would certainly want to use multimodular and Chinese remainder techniques in order to avoid the growth of the integers in the intermediate computations; this would require to extend our result to finite fields.

This naturally opens the more general question of fast computation with multivariate power series over any ring. The point is to extend Lemma 1; the main difficulty is to choose points in the base ring such that distinct monomials take distinct values on these points. A first result in this direction, based on combinatorial arguments, is given by Zippel [1990].

Acknowledgments

We greatly thank B. Salvy for his useful comments, O. Ruatta for sending typing mistakes to us and J. van der Hoeven for his pointing out to us an error in the complexity estimate in an earlier version of this paper.

References

M. Ben-Or and P. Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation. In *20th Annual ACM Symposium Theory of Computing*, 1988.

F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. In *Proceedings ISSAC'95*, pages 158–166. ACM Press, 1995.

R. P. Brent and H. T. Kung. Fast algorithms for composition and reversion of multivariate power series. In *Proceedings of the Conference on Theoretical Computer Science, University of Waterloo, Ontario, Canada*, pages 149–158, 1977.

P. Bürgisser, M. Clausen, and M. A. Shokrolahi. *Algebraic Complexity Theory.* Springer, 1997.

J. Canny, E. Kaltofen, and Y. Lakshman. Solving systems of non-linear polynomial equations faster. In *Proceedings ISSAC'89*, pages 121–128. ACM Press, 1989.

M. Giusti, G. Lecerf, and B. Salvy. A Gröbner free alternative for polynomial system solving. *Journal of Complexity*, 17(1):154–211, 2001.

D. Y. Grigoriev and M. Karpinski. The matching problem for bipartite graphs with polynomially bounded permanents is in NC. In *Proceedings of the 28th IEEE Symposium on the Foundations of Computer Science*, pages 166–172, 1987.

J. van der Hoeven. Relax, but don't be too lazy. Technical report, Université de Paris-Sud, Mathématiques, Bâtiment 425, F-91405 Orsay, 2001.

L. Kronecker. Grundzüge einer arithmetischen Theorie der algebraischen Grössen. *Journal für die Reine und Angewandte Mathematik*, 92:1–122, 1882.

G. Lecerf. Quadratic Newton iterator for systems with multiplicity. *Journal of FoCM (to appear)*, 2001a.

G. Lecerf. *Une alternative aux méthodes de réécriture pour la résolution des systèmes algébriques*. PhD thesis, École polytechnique, 2001b.

A. Péladan. *Tests effectifs de nullité dans des extensions d'anneaux différentiels*. PhD thesis, École polytechnique, 1997.

A. Schönhage. Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2. *Acta Informatica*, 7:395–398, 1977.

A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.

É. Schost. *Sur la résolution des systèmes polynomiaux à paramètres*. PhD thesis, École polytechnique, http://www.gage.polytechnique.fr/schost.html, 2000.

P. Tiwari. Parallel algorithms for instance of the linear matroid parity with a small number of solutions. Technical Report IBM Reseach Report RC 12766, IBM Watson Research Center, Yorktown Heights, NY, 1987.

R. Zippel. Interpolating polynomials from their values. *Journal of Symbolic Computation*, 9(3):375–403, 1990.