

# Fast Multivariate Power Series Multiplication in Characteristic Zero

Grégoire Lecerf

GAGE, École polytechnique (France)

June 11, 2001

Summary by Ludovic Meunier

## Abstract

Let  $S$  be a multivariate power series ring over a field of characteristic zero. The article [5] presents an asymptotically fast algorithm for multiplying two elements of  $S$  truncated according to *total* degree. Up to logarithmic factors, the complexity of the algorithm is optimal, in the sense that it is linear in the size of the output.

## 1. Introduction

Let  $k$  be a field of characteristic zero. We write  $S = k[[x_1, \dots, x_n]]$  for the multivariate power series ring in the  $n$  variables  $x_1, \dots, x_n$ . Let  $\mathfrak{J}$  be any ideal of  $S$ . By computing at *precision*  $\mathfrak{J}$  in  $S$ , we understand computing modulo the ideal  $\mathfrak{J}$  in  $S$ . In other words, power series in  $S$  are regarded as vectors in the  $k$ -algebra  $S/\mathfrak{J}$ . We denote by  $\mathfrak{m}$  the maximal ideal  $(x_1, \dots, x_n)$  in  $S$  and by  $d$  any positive integer. The paper [5] sets the problem of a fast algorithm for multiplying two power series in  $S$  truncated in *total* degree  $d$ , that is computed at precision  $\mathfrak{m}^{d+1}$ .

The general question of a fast algorithm for multivariate multiplication in  $S$  modulo *any* ideal remains an open problem and has received very little attention in the literature. Previous works (e.g., [2]) investigated computation modulo the ideal  $(x_1^{d+1}, \dots, x_n^{d+1})$ , that is truncation according to *partial* degree with respect to each variable  $x_i$ . The method used is called Kronecker's substitution and is briefly discussed in Section 3.

The need for multiplication routines modulo  $\mathfrak{m}^{d+1}$  arises in various fields, such as polynomial system solving [7] and treatment of systems of partial differential equations.

The efficiency of the algorithm is measured with respect to the model of *nonscalar* complexity. By nonscalar complexity, we understand the number of primitive operations in the field  $k$  needed to complete the algorithm, independently of the sizes of the numbers involved (see [3]). We now introduce some notation. We denote by  $D = \deg(\mathfrak{m}^{d+1})$  the *degree* of the ideal  $\mathfrak{m}^{d+1}$ .  $D$  is the number of monomials in  $S$  which are not in  $\mathfrak{m}^{d+1}$ , that is the dimension of the  $k$ -algebra  $S/\mathfrak{m}^{d+1}$ . Simple combinatorial considerations give

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

We set  $C := \deg(\mathfrak{m}^d)$  and denote by  $\mathcal{M}_u(\delta)$  the complexity of the multiplication of two univariate polynomials of degree  $\delta$  in  $k[t]$ .

The next section presents the algorithm; its complexity belongs to

$$(1) \quad \mathcal{O}(D \log^3 D \log \log D).$$

Since  $D$  is the size of the output, the algorithm is optimal, up to the logarithmic factors.

## 2. The Algorithm

**2.1. Description.** The first step of the algorithm consists in translating the multivariate problem into a univariate one. This is motivated by the fact that fast algorithms for univariate power series multiplication are known (e.g., [6]).

Let  $t$  be a new variable. We consider the substitution

$$\begin{aligned} \tilde{\mathcal{R}}_t : S/\mathfrak{m}^{d+1} &\longrightarrow k[x_1, \dots, x_n][[t]]/(t^{d+1}) \\ f(x_1, \dots, x_n) &\longmapsto f(x_1t, \dots, x_nt). \end{aligned}$$

If  $f$  is an element of  $S/\mathfrak{m}^{d+1}$ ,  $\tilde{\mathcal{R}}_t(f)$  is a univariate power series in the *single* variable  $t$  truncated at degree  $d$ . It can then be written  $\tilde{\mathcal{R}}_t(f) = \tilde{f}_0 + \tilde{f}_1t + \dots + \tilde{f}_dt^d$ , where each coefficient  $\tilde{f}_i$  is a homogeneous multivariate polynomial in the variables  $x_1, \dots, x_n$  of *total* degree  $i$ . This remark on the degree suggests that:

1. the substitution  $\tilde{\mathcal{R}}_t$  is optimal, in the sense that it provides us with a representation of  $f$  that retains exactly the monomials that form a basis of  $S/\mathfrak{m}^{d+1}$ . In particular, the algorithm does not suffer from any overhead caused by unnecessary terms (see Section 3);
2. in view of the homogeneity of the  $\tilde{f}_i$ , keeping all of the variables  $x_i$  is redundant. The substitution defined by

$$\begin{aligned} \mathcal{R}_t : S/\mathfrak{m}^{d+1} &\longrightarrow k[x_2, \dots, x_n][[t]]/(t^{d+1}) = (k[[t]]/(t^{d+1})) [x_2, \dots, x_n] \\ f(x_1, \dots, x_n) &\longmapsto f(t, x_2t, \dots, x_nt) \end{aligned}$$

reduces the complexity in the step of *evaluation-interpolation* (see below):  $n - 1$  variables, instead of  $n$  variables, are actually needed.

The second step of the algorithm performs the multiplication. Let  $f$  and  $g$  be two power series in  $S/\mathfrak{m}^{d+1}$  and  $h$  be the product  $fg$  in  $S/\mathfrak{m}^{d+1}$ . The equality  $h = fg$  turns into

$$(2) \quad \mathcal{R}_t(h) = \mathcal{R}_t(f)\mathcal{R}_t(g).$$

Consequently, we concentrate on a fast way to compute  $\mathcal{R}_t(h)$ . We use an evaluation-interpolation scheme. We first consider the *evaluation* map at the point  $P = (p_2, \dots, p_n)$  in  $k^{n-1}$  defined by

$$\begin{aligned} \mathcal{E}_P : (k[[t]]/(t^{d+1})) [x_2, \dots, x_n] &\longrightarrow k[[t]]/(t^{d+1}) \\ f(x_2, \dots, x_n) &\longmapsto f(P). \end{aligned}$$

We then apply  $\mathcal{E}_P$  to equation (2), which yields

$$(3) \quad \mathcal{E}_P(\mathcal{R}_t(h)) = \mathcal{E}_P(\mathcal{R}_t(f))\mathcal{E}_P(\mathcal{R}_t(g)) \pmod{t^{d+1}}.$$

Equation (3) holds for any point  $P$  and computes the product  $\mathcal{R}_t(h)$  at  $P$  by using a *univariate* power series multiplication algorithm. Such an algorithm is described in [6].

The last step of the algorithm consists in reconstructing  $h$  from a set of values of  $\mathcal{R}_t(h)$ . We regard  $\mathcal{R}_t(h)$  as a multivariate polynomial in the variables  $x_2, \dots, x_n$ . There exists an *interpolation* map

$$\begin{aligned} \mathcal{I} : (k[[t]]/(t^{d+1}))^C &\longrightarrow (k[[t]]/(t^{d+1})) [x_2, \dots, x_n] \\ (f(P_1), \dots, f(P_C)) &\longmapsto f(x_2, \dots, x_n), \end{aligned}$$

which recovers  $\mathcal{R}_t(h)$  from a set of  $C$  pairwise distinct values  $\{\mathcal{E}_{P_1}(\mathcal{R}_t(h)), \dots, \mathcal{E}_{P_C}(\mathcal{R}_t(h))\}$ . The evaluation points  $P_i$ , for  $i$  in  $1, \dots, C$ , are chosen to be powers of distinct prime numbers, namely  $P_i = (p_2^i, \dots, p_n^i)$ , where  $p_j$  are distinct prime numbers. Note the key point is that the characteristic of the ground field  $k$  is zero, so that all  $\mathcal{E}_{P_i}(\mathcal{R}_t(h))$  have pairwise distinct values. An implementation

of both maps  $\mathcal{E}_P$  and  $\mathcal{I}$  is described by J. Canny, E. Kaltofen, and Y. Lakshman in [4]. Their method relies on fast univariate multipoint evaluation and interpolation (e.g., [1]).

Finally, we reconstruct  $h$  from  $\mathcal{R}_t(h)$ . If  $\mathcal{R}_t(h) = h_0 + h_1 t + \cdots + h_d t^d$  is given,  $h$  is obtained by homogenizing each  $h_i$  in degree  $i$  with respect to the variable  $x_1$  and then evaluating at  $t = 1$ .

We are now ready to unfold the algorithm.

```

MultivariatePS_Mult := proc(f,g)
  (1)   $F \leftarrow \mathcal{R}_t(f); G \leftarrow \mathcal{R}_t(g);$            // new representation
  (2)  for  $i$  in  $(P_1, \dots, P_C)$  do                     // evaluation
         $F_{P_i} \leftarrow \mathcal{E}_{P_i}(F); G_{P_i} \leftarrow \mathcal{E}_{P_i}(G);$ 
  (3)  for  $i$  to  $C$  do                                     // univariate multiplication
         $H_{P_i} \leftarrow F_{P_i} G_{P_i};$ 
  (4)   $\mathcal{R}_t(h) \leftarrow \mathcal{I}(H_{P_1}, \dots, H_{P_C});$        // interpolation
  (5)   $h \leftarrow$  homogenization in degree with respect to  $x_1$  // reconstruction
        in  $\mathcal{R}_t(h);$ 
  return  $h;$ 

```

The next section derives the complexity result claimed by (1).

**2.2. Complexity.** Steps 1 and 5 can be performed in  $\mathcal{O}(C)$  operations. We examine the cost of Steps 2, 3, and 4 separately:

- Step 2 evaluates the  $d$  coefficients of  $F$  and  $G$  at  $C$  points. The  $C$  points  $P_i$  are chosen to be powers of the  $n - 1$  distinct prime numbers  $(p_2, \dots, p_n)$ , namely  $P_i = (p_2^i, \dots, p_n^i)$ . Each coefficient can be computed in  $\mathcal{O}(\mathcal{M}_u(C) \log C)$  operations, according to the algorithm for fast multipoint evaluation given in [4]. This yields an overall complexity of  $\mathcal{O}(d\mathcal{M}_u(C) \log C)$  for Step 2.
- Step 3 performs  $C$  univariate power series products. Each multiplication requires  $\mathcal{O}(\mathcal{M}_u(d))$  operations. Complexity of Step 3 is then  $\mathcal{O}(C\mathcal{M}_u(d))$ .
- Step 4 interpolates the  $d$  coefficients of  $H$ . Each interpolation requires  $\mathcal{O}(\mathcal{M}_u(C) \log C)$  operations, also using the algorithm presented in [4]. Step 4 then requires  $\mathcal{O}(d\mathcal{M}_u(C) \log C)$  operations.

The overall complexity of the algorithm is then derived by replacing  $\mathcal{M}_u(C)$  by its estimate  $\mathcal{O}(C \log C \log \log C)$  obtained in [6] and noting that  $C < D \log(D)/d$ . This yields

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

**2.3. Generalization.** We mention that van der Hoeven generalized the algorithm to the case when

$$\mathfrak{J} = (x_1^{\alpha_1} \dots x_n^{\alpha_n}, \quad \text{for } \alpha_1 d_1 + \cdots + \alpha_n d_n > d),$$

where the  $\alpha_i$  are positive integers, by using the substitution defined by

$$\begin{aligned} \mathcal{V}_t : S/\mathfrak{J} &\longrightarrow k[x_2, \dots, x_n][[t]]/(t^{d+1}) \\ f(x_1, \dots, x_n) &\longmapsto f(t^{\alpha_1}, x_2 t^{\alpha_2}, \dots, x_n t^{\alpha_n}) \end{aligned}$$

instead of  $\mathcal{R}_t$ . The rest of the algorithm remains unaltered.

### 3. Appendix: Kronecker's Substitution

Kronecker's substitution is defined by the map

$$\begin{aligned} \mathcal{K}_t : S/\mathfrak{J} &\longrightarrow k[[t]]/t^{(2d+1)^n} \\ f(x_1, \dots, x_n) &\longmapsto f(t, t^{2d+1}, \dots, t^{(2d+1)^{n-1}}), \end{aligned}$$

where  $\mathfrak{J} = (x_1^{d+1}, \dots, x_n^{d+1})$ . This substitution truncates power series in *partial* degree  $d$  with respect to each variable  $x_i$ . Let  $f$  be a power series in  $S/\mathfrak{J}$ , one recovers the coefficient of  $x_1^{e_1} \dots x_n^{e_n}$  in  $f$  by simply reading off the coefficient of  $t^{e_1 + (2d+1)e_2 + \dots + (2d+1)^{n-1}e_n}$  in  $\mathcal{K}_t(f)$ . The cost of this algorithm is the cost of the multiplication of two univariate polynomials of degree  $(2d)^n$ , that is  $\mathcal{O}(\mathcal{M}_u((2d)^n))$ . This is the lowest known complexity for multivariate power series multiplication modulo the ideal  $(x_1^{d+1}, \dots, x_n^{d+1})$ . In particular, when addressed in this context, the algorithm presented above requires precision  $\mathfrak{m}^{nd+1}$  and yields a similar complexity.

Kronecker's substitution may be used to compute modulo  $\mathfrak{m}^{d+1}$  as well. However, it results in a significant overhead of  $\mathcal{O}(2^n n!)$ , for fixed  $n$  and  $d \gg n$ , with respect to the size of the power series.

#### Bibliography

- [1] Aho (Alfred V.), Hopcroft (John E.), and Ullman (Jeffrey D.). – *The design and analysis of computer algorithms*. – Addison-Wesley Publishing Co., Reading, Mass.-London-Amsterdam, 1975, x+470p. Second printing, Addison-Wesley Series in Computer Science and Information Processing.
- [2] Brent (R. P.) and Kung (H. T.). – Fast algorithms for composition and reversion of multivariate power series (preliminary version). In *Proceedings of a Conference on Theoretical Computer Science Department of Computer Science, University of Waterloo, Waterloo, Ontario (August 1977)*, pp. 149–158. – 1977.
- [3] Bürgisser (Peter), Clausen (Michael), and Shokrollahi (M. Amin). – *Algebraic complexity theory*. – Springer-Verlag, Berlin, 1997, xxiv+618p. With the collaboration of Thomas Lickteig.
- [4] Canny (John F.), Kaltofen (Erich), and Lakshman Yagati. – Solving systems of nonlinear polynomial equations faster. In Gonnet (Gaston) (editor), *Symbolic and Algebraic Computation (International Symposium ISSAC'89, Portland, Oregon, USA, July 17-19, 1989)*, pp. 121–128. – ACM Press, 1989. Conference proceedings.
- [5] Lecerf (Grégoire) and Schost (Éric). – Fast multivariate power series multiplication in characteristic zero. – Available from <http://www.medicis.polytechnique.fr/~schost/>, 2001.
- [6] Schönhage (A.). – Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2. *Acta Informatica*, vol. 7, n° 4, 1976/77, pp. 395–398.
- [7] Schost (Éric). – *Sur la résolution des systèmes polynomiaux à paramètres*. – PhD thesis, École polytechnique, Palaiseau, France, January 2001. Defended on December 7, 2000.