Faster relaxed multiplication

Joris van der Hoeven

Faster relaxed multiplication*

by Joris van der Hoeven

CNRS, Laboratoire d'informatique
École polytechnique
91128 Palaiseau Cedex
France
Email: joris@texmacs.org

April 29, 2014

Abstract

In previous work, we have introduced several fast algorithms for relaxed power series multiplication (also known under the name on-line multiplication) up to a given order n. The fastest currently known algorithm works over an effective base field $\mathbb{K}$ with sufficiently many 2^p-th roots of unity and has algebraic time complexity $\mathcal{O}(n \log n\, e^{2\sqrt{\log 2}\sqrt{\log\log n}})$. In this paper, we will generalize this algorithm to the cases when $\mathbb{K}$ is replaced by an effective ring of positive characteristic or by an effective ring of characteristic zero, which is also torsion-free as a $\mathbb{Z}$-module and comes with an additional algorithm for partial division by integers. In particular, we may take $\mathbb{K}$ to be any effective field. We will also present an asymptotically faster algorithm for relaxed multiplication of p-adic numbers.

Keywords: power series, multiplication, on-line algorithm, FFT, computer algebra

A.M.S. subject classification: 68W30, 30B10, 68W25, 33F05, 11Y55, 42-04

1 Introduction

1.1 Relaxed resolution of recursive equations

Let $\mathbb{A}$ be an effective (possibly non-commutative) ring; i.e., we assume data structures for representing the elements of $\mathbb{A}$ and algorithms for performing the ring operations $+$, $-$ and $\times$. The aim of algebraic complexity theory is to study the cost of basic or more complex algebraic operations over $\mathbb{A}$ (such as the multiplication of polynomials or matrices) in terms of the number of operations in $\mathbb{A}$.

The algebraic complexity usually does not coincide with the bit complexity, which also takes into account the potential growth of the actual coefficients in $\mathbb{A}$. Nevertheless, understanding the algebraic complexity usually constitutes a first useful step towards understanding the bit complexity. Of course, in the special case when $\mathbb{A}$ is a finite field, both complexities coincide up to a constant factor.

One of the most central operations is polynomial multiplication. We will denote by $\mathsf{M}_{\mathbb{A}}(n)$ the number of operations required to multiply two polynomials of degrees $<n$ in $\mathbb{A}[x]$. If $\mathbb{A}$ admits primitive 2^p-th roots of unity for all p, then we have $\mathsf{M}_{\mathbb{A}}(n) = \mathcal{O}(n \log n)$ using FFT multiplication, which is based on the fast Fourier transform [12]. In general, it has been shown [28, 10] that $\mathsf{M}_{\mathbb{A}}(n) = \mathcal{O}(n \log n \log\log n)$. The complexities of most other operations (division, Taylor shift, extended g.c.d., multipoint evaluation, interpolation, etc.) can be expressed in terms of $\mathsf{M}_{\mathbb{A}}(n)$. Often, the cost of such other operations is simply $\tilde{\mathcal{O}}(\mathsf{M}_{\mathbb{A}}(n)) = \tilde{\mathcal{O}}(n)$, where $\tilde{\mathcal{O}}(T(n))$ stands for $\mathcal{O}(T(n)\ (\log T(n))^{\mathcal{O}(1)})$; see [2, 6, 15] for some classical results along these lines.

The complexity of polynomial multiplication is fundamental for studying the cost of operations on formal power series in $\mathbb{A}[[z]]$ up to a given truncation order n. Clearly, it is possible to perform the multiplication up to order n in time $\mathcal{O}(\mathsf{M}_{\mathbb{A}}(n))$: it suffices to multiply the truncated power series at order n and truncate the result. Using Newton's method, and assuming that $\mathbb{Q} \subseteq \mathbb{A}$, it is also possible to compute exp, sin, etc. up to order n in time $\mathcal{O}(\mathsf{M}_{\mathbb{A}}(n))$. More generally, it has been shown in [9, 17, 29, 23] that the power series solutions of algebraic differential equations with coefficients in $\mathbb{A}[[z]]$ can be computed up to order n in time $\mathcal{O}(\mathsf{M}_{\mathbb{A}}(n))$. However, in this case, the "$\mathcal{O}$" hides a non-trivial constant factor which depends on the expression size of the equation that one wants to solve.

The *relaxed* approach for computations with formal power series makes it possible to solve equations in quasi-optimal time with respect to the sparse expression size of the equations. The idea is to consider power series $f \in \mathbb{A}[[z]]$ as streams of coefficients f_0, f_1, ... and to require that all operations are performed "without delay" on these streams. For instance, for a multiplication $h = f\,g$ of two power series f, $g \in \mathbb{A}[[z]]$, we require that h_n is computed *as soon as* $f_0, g_0, \ldots, f_n, g_n$ are known. Any algorithm which has this property will be called a *relaxed* or *on-line* algorithm for multiplication.

Given a relaxed algorithm for multiplication, it is possible to let the later coefficients $f_{n+1}, g_{n+1}, f_{n+2}, g_{n+2}, \ldots$ of the input *depend* on the known coefficients h_0, ..., h_n of the output. For instance, given a power series $f \in \mathbb{A}[z]$ with $f_0 = 0$, we may compute $g = \exp f$ using the formula

$$g = \int f' g, \tag{1}$$

*. This work has been partly supported by the French ANR-09-JCJC-0098-01 MaGiX project, and by the Digiteo 2009-36HD grant of the Région Ile-de-France.

provided that $\mathbb{Q} \subseteq \mathbb{A}$. Indeed, extraction of the coefficient of z^n in g and $\int f' g$ yields

$$g_n = \frac{1}{n} (f' g)_{n-1},$$

and $(f' g)_{n-1}$ only depends on $g_0, ..., g_{n-1}$. More generally, we define an equation of the form

$$f = \Phi(f) \tag{2}$$

to be *recursive*, if $\Phi(f)_n$ only depends on $f_0, ..., f_{n-1}$. Replacing $\mathbb{A}$ by $\mathbb{A}^r$, we notice that the same terminology applies to systems of r equations. In the case of an implicit equation, special rewriting techniques can be implied in order to transform the input equation into a recursive equation [24, 22, 5].

Let $\mathsf{R}_{\mathbb{A}}(n)$ denote the cost of performing a relaxed multiplication up to order n. If Φ is an expression which involves s multiplications and t other "linear time" operations (additions, integrations, etc.), then it follows that (2) can be solved up to order n in time $\mathcal{O}(s\, \mathsf{R}_{\mathbb{A}}(n) + t\, n)$. If we had $\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}(\mathsf{M}_{\mathbb{A}}(n))$, then this would yield an optimal algorithm for solving (2) in the sense that the computation of the solution would essentially require the same time as its verification.

1.2 Known algorithms for relaxed multiplication

The naive $\mathcal{O}(n^2)$ algorithm for computing $h = f\, g$, based on the formula

$$h_n = f_0\, g_n + f_1\, g_{n-1} + \cdots + f_n\, g_0,$$

is clearly relaxed. Unfortunately, FFT multiplication is *not* relaxed, since $h_0, ..., h_n$ are computed simultaneously as a function of $f_0, g_0, ..., f_n, g_n$, in this case.

In [16, 17] it was remarked that Karatsuba's $\mathcal{O}(n^{\log 3/\log 2})$ algorithm [25] for multiplying polynomials can be rewritten in a relaxed manner. Karatsuba multiplication and its relaxed version thus require *exactly* the same number of operations. In [16, 17], an additional fast relaxed algorithm was presented with time complexity

$$\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}(\mathsf{M}_{\mathbb{A}}(n) \log n). \tag{3}$$

We were recently made aware of the fact that a similar algorithm was first published in [13]. However, this early paper was presented in a different context of online (relaxed) multiplication of integers (instead of power series), and without the application to the resolution of recursive equations (which is quite crucial from our perspective).

An interesting question remained: can the bound (3) be lowered further, be it by a constant factor? In [18], it was first noticed that an approximate factor of two can be gained if one of the multiplicands is known beforehand. For instance, if we want to compute $g = \exp f$ for a known series f with $f_0 = 0$, then the coefficients of f' are already known in the product $f' g$ in (1), so only one of the inputs depends on the output. An algorithm for the computation of $h = f\, g$ is said to be *semi-relaxed*, if h_n is written to the output as soon as $f_0, ..., f_n$ are known, but all coefficients of g are known beforehand. We will denote by $\mathsf{S}_{\mathbb{A}}(n)$ the complexity of semi-relaxed multiplication. We recall from [21] (see also Section 3) that relaxed multiplication reduces to semi-relaxed multiplication:

$$\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}(\mathsf{S}_{\mathbb{A}}(n)).$$

It has recently been pointed out [26] that the factor 2 that was gained in the case of semi-relaxed multiplication can also be gained in the general case.

The first reduction of (3) by a non-constant factor was published in [21], and uses the technique of *FFT blocking* (which has also been used for the multiplication of multivariate polynomials and power series in [17, Section 6.3], and for speeding up Newton iterations in [3, 23]). Under the assumption that $\mathbb{A}$ admits primitive 2^k-th roots of unity for all k (or at least for all k with $2^k \leqslant n$), we showed that

$$\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}(n \log n\, \mathrm{e}^{2\sqrt{\log 2}\sqrt{\log\log n}}). \tag{4}$$

The function $\mathrm{e}^{2\sqrt{\log 2}\sqrt{\log\log n}}$ has slower growth than any strictly positive power of $\log n$. It is convenient to write $F(n) = \mathcal{O}^{\flat}(T(n))$ whenever $F(n) = \mathcal{O}(T(n)\, (\log T(n))^{\alpha})$ for all $\alpha > 0$. In particular, it follows that

$$\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}^{\flat}(n \log n).$$

In Section 3, we will recall the main ideas from [21] which lead to the complexity bound (4).

In fact, in Section 4, we will see that the complexity bound from [21] can be further reduced to

$$\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}(n \log n\, \mathrm{e}^{\sqrt{2\log 2}\sqrt{\log\log n}} \sqrt{\log\log n}).$$

Since such complexity bounds involve rather complex expressions, it will be convenient to use the following abbreviations:

$$\begin{aligned} \mathsf{R}_*(n) &= n \log n\, \mathrm{e}^{\sqrt{2\log 2}\sqrt{\log\log n}} \sqrt{\log\log n} \\ \mathsf{R}_{**}(n) &= \mathsf{R}_*(n) \log\log n \\ \mathsf{R}_{***}(n) &= \mathsf{R}_*(n)\, (\log\log n)^2 \log\log\log n. \end{aligned}$$

Clearly, $\mathsf{R}_*(n) = \mathcal{O}(\mathsf{R}_{**}(n)) = \mathcal{O}(\mathsf{R}_{***}(n)) = \mathcal{O}^{\flat}(n \log n)$.

1.3 Improved complexity bounds

We recall that the characteristic of a ring $\mathbb{A}$ is the integer $k \in \mathbb{N}$ such that the canonical ring homomorphism $\mathbb{Z} \to \mathbb{A}$ has kernel $k\, \mathbb{Z}$. If $\mathbb{A}$ is torsion-free as a $\mathbb{Z}$-module (i.e. $k\, x = 0 \Rightarrow x = 0$ for any $k \in \mathbb{Z} \setminus \{0\}$ and $x \in \mathbb{A}$), then we will say that $\mathbb{A}$ admits an effective partial division by integers if there exists an algorithm which takes $k \in \mathbb{Z} \setminus \{0\}$ and $x \in k\, \mathbb{A}$ on input and which returns the unique $y \in \mathbb{A}$ with $x = k\, y$ on output. We will count a unit cost for such divisions. The main result of this paper is:

Theorem 1. *Assume that one of the following two holds:*

- $\mathbb{A}$ *is an effective ring of characteristic zero, which is torsion-free as a* $\mathbb{Z}$*-module, and which admits an effective partial division by integers.*
- $\mathbb{A}$ *is an effective ring of positive characteristic.*

Then we have

$$\mathsf{R}_{\mathbb{A}}(n) = \mathcal{O}^{\flat}(n \log n). \tag{5}$$

We notice that the theorem holds in particular if $\mathbb{A}$ is an effective field. In Section 8, we will also consider the relaxed multiplication of p-adic numbers, with $p \in \mathbb{N}$ and $p \geqslant 2$. If we denote by $\mathsf{I}(k)$ the bit complexity of multiplying two k-bit integers, then [13] essentially provided an

algorithm of bit complexity $\mathcal{O}(\mathsf{I}(n)\log n)$ for the relaxed multiplication of 2-adic numbers at order $\mathcal{O}(2^n)$. Various algorithms and benchmarks for more general p were presented in [4]. It is also well known [33, 11, 28, 14] that $\mathsf{I}(n)=\mathcal{O}^\flat(n\log n)$. Let $\mathsf{R}_p(n)$ denote the bit complexity of the relaxed multiplication of two p-adic numbers modulo p^n. In Section 8, we will prove the following new result:

Theorem 2. *Let $p\in\mathbb{N}$ with $p\geqslant 2$. Then we have*

$$\mathsf{R}_p(n) \;=\; \mathcal{O}^\flat(n\log n\log p\log\log p),$$

uniformly in n and p.

For comparison, the best previously known bound for relaxed multiplication in $\mathbb{Z}_p$ was

$$\begin{aligned}\mathsf{R}_p(n) &= \mathcal{O}(\mathsf{I}(n(\log p+\log n))\log n)\\ &= \begin{cases}\mathcal{O}^\flat(n\log^3 n) & \text{if } p=\mathcal{O}(n)\\ \mathcal{O}^\flat(n\log^2 n\log p) & \text{if } n=\mathcal{O}(p)\end{cases}\end{aligned}$$

We thus improved the previous bound by a factor $\log n/\log\log p$ at least, up to sublogarithmic terms.

The main idea which allows for the present generalizations is quite straightforward. In our original algorithm from [21], the presence of sufficiently many primitive 2^k-th roots of unity in $\mathbb{A}$ gives rise to a quasi-optimal evaluation-interpolation strategy for the multiplication of polynomials. More precisely, given two polynomials of degrees $<n$, their FFT-multiplication only requires $\mathcal{O}(n)$ evaluation and interpolation points, and both the evaluation and interpolation steps can be performed efficiently, using only $\mathcal{O}(n\log n)$ operations. Now it has recently been shown [8] that quasi-optimal evaluation-interpolation strategies still exist if we evaluate and interpolate at points in geometric progressions instead of roots of unity. This result is the key to our new complexity bounds, although further technical details have to be dealt with in order to make things work for various types of effective rings $\mathbb{A}$. We also notice that the main novelty of [8] concerns the interpolation step. Fast evaluation at geometric progressions was possible before using the so called *chirp transform* [27, 7]. For effective rings $\mathbb{A}$ of positive characteristic, this would actually have been sufficient for the proving the bound (5).

Our paper is structured as follows. Since the algorithms of [8] were presented in the case when $\mathbb{A}$ is an effective field, Section 2 starts with their generalization to more general effective rings $\mathbb{A}$. These generalizations are purely formal and contain no essentially new ideas. In Section 3, we give a short survey of the algorithm from [21], but we recommend reading the original paper for full technical details. In Section 4, we sharpen the complexity analysis for the algorithm from [21]. In Section 5, we prove Theorem 1 in the case when $\mathbb{A}$ has characteristic zero. In Section 6, we turn our attention to the case when $\mathbb{A}$ has prime characteristic p. If the characteristic is sufficiently large, then we may find sufficiently large geometric progressions in $\mathbb{A}\cap\mathbb{Z}$ in order to generalize the results from Section 5. Otherwise, we have to work over $\mathbb{A}\otimes\mathbb{F}_{p^k}$ for some sufficiently large k. In Section 7, we complete the proof of Theorem 1; the case when the characteristic is a prime power is a refinement of the result from Section 6. The remaining case is done *via* Chinese remaindering. In Section 8, we will prove Theorem 2.

Acknowledgments. I am grateful to the referees for their detailed comments and suggestions. A special thanks goes to the referee who made a crucial suggestion for the improved complexity analysis in Section 4.

2 Multipoint evaluation and interpolation

Let $\mathbb{D}$ be a (commutative) effective integral domain and let $\mathbb{K}$ be its quotient field. Assume that $\mathbb{K}$ has characteristic zero. Let $\mathbb{M}$ be an effective torsion-free $\mathbb{D}$-module and $\mathbb{V}=\mathbb{K}\otimes_{\mathbb{D}}\mathbb{M}$. Elements of $\mathbb{K}$ and $\mathbb{V}$ are fractions x/s with $x\in\mathbb{D}$ (resp. $x\in\mathbb{M}$) and $s\in\mathbb{D}\setminus\{0\}$, and the operations $-$, $+$, $\times$ on such fractions are as usual:

$$\begin{aligned}-\frac{x}{s} &= \frac{-x}{s}\\ \frac{x}{s}+\frac{y}{t} &= \frac{t\,x+s\,y}{s\,t}\\ \frac{x}{s}\,\frac{y}{t} &= \frac{x\,y}{s\,t}\end{aligned}$$

For $x/s\in\mathbb{K}^*$, we also have $(x/s)^{-1}=s/x$. It follows that $\mathbb{K}$ is an effetive field and $\mathbb{V}$ an effective $\mathbb{K}$-vector space. Moreover, all field operations in $\mathbb{K}$ (and all vector space operations in $\mathbb{V}$) can be performed using only $\mathcal{O}(1)$ operations in $\mathbb{D}$ (resp. $\mathbb{D}$ or $\mathbb{M}$).

We will say that $\mathbb{M}$ admits an effective partial division, if for every $s\in\mathbb{D}\setminus\{0\}$ and $x\in s\,\mathbb{M}$, we can compute the unique $y\in\mathbb{M}$ with $x=s\,y$. In that case, and we will count any division of the above kind as one operation in $\mathbb{M}$. Similarly, given a fixed $s\in\mathbb{D}\setminus\{0\}$, we say that $\mathbb{M}$ admits an effective partial division by $sx\in s\,\mathbb{M}$, we can compute the unique $y\in\mathbb{M}$ with $x=s\,y$. Given $n\in\mathbb{N}$, we define

$$\mathbb{M}[z]_n \;=\; \{P\in\mathbb{M}[z]\colon \deg P<n\}.$$

Given $P\in\mathbb{D}[z]_n$ and $Q\in\mathbb{M}[z]_n$, we will denote by $\mathsf{M}_{\mathbb{M}}(n)$ the number of operations in $\mathbb{D}$ and $\mathbb{M}$ which are needed in order to compute the product $PQ\in\mathbb{M}[z]$.

Lemma 3. *Let $\mathbb{D}$ be an effective integral domain and $\mathbb{M}$ an effective torsion-free $\mathbb{D}$-module. There exists a constant K, such that the following holds: for any $n>0$, $P\in\mathbb{M}[z]_n$ and $q\in\mathbb{D}\setminus\{0\}$ such that $1, q, ..., q^{n-1}$ are pairwise distinct, and such that $\mathbb{D}$ admits effective divisions by q and $q-1, q^2-1, ..., q^{n-1}-1$, we have:*

a) We may compute $P(1)$, ..., $P(q^{n-1})$ from P using $K\,\mathsf{M}_{\mathbb{M}}(n)$ operations in $\mathbb{D}$ and $\mathbb{M}$.

b) We may reconstruct P from $P(1),...,P(q^{n-1})$ using $K\,\mathsf{M}_{\mathbb{M}}(n)$ operations in $\mathbb{D}$ and $\mathbb{M}$.

Proof. In the case when $\mathbb{D}=\mathbb{K}$ is a field and $\mathbb{M}=\mathbb{V}=\mathbb{K}$, this result was first proven in [8]. More precisely, the conversions can be done using the algorithms **NewtonEvalGeom**, **NewtonInterpGeom**, **NewtonToMonomialGeom** and **MonomialToNewtonGeom** in that paper. Examining these algorithms, we observe that general elements in $\mathbb{D}$ are only multiplied with elements in $\mathbb{Z}[q]$ and divided by elements of the set $\{q, q-1, q^2-1, ..., q^{n-1}-1\}$. In particular, the algorithms can still be applied in the more general case when $\mathbb{D}=\mathbb{K}$ is a field and $\mathbb{M}=\mathbb{V}$ a vector space.

If $\mathbb{D}$ is only an effective integral domain and $\mathbb{M}$ an effective torsion-free $\mathbb{D}$-module with an effective partial division, then we define the effective field $\mathbb{K}$ and the effective vector space $\mathbb{V}$ as above, and we may still apply the generalized algorithms for multipoint evaluation and interpolation in $\mathbb{V}$. In particular, both multipoint evaluation and interpolation can still be done using $\mathcal{O}(\mathsf{M}_{\mathbb{V}}(n))$ operations in $\mathbb{K}$ and $\mathbb{V}$, whence $\mathcal{O}(\mathsf{M}_{\mathbb{M}}(n))$ operations in $\mathbb{D}$ and $\mathbb{M}$. If we *know* that the end-results of these algorithms are really in the subspace $\mathbb{M}^n$ of $\mathbb{V}^n$ (or in the submodule $\mathbb{M}[z]_n$ of $\mathbb{V}[z]_n$), then we use the partial division in $\mathbb{M}$ to replace their representations in $\mathbb{V}^n$ (or $\mathbb{V}[z]_n$) by representations in $\mathbb{M}^n$ (or $\mathbb{M}[z]_n$). □

3 Survey of blockwise relaxed multiplication

Let $\mathbb{A}$ be an effective (possibly non-commutative) ring and recall that

$$\mathbb{A}[z]_n \;=\; \{P \in \mathbb{A}[z] : \deg P < n\}.$$

Given a power series $f \in \mathbb{A}[[z]]$ and $i < j$, we will also use the notations

$$\begin{aligned} f_{i;j} &= f_i\, z^i + \cdots + f_{j-1}\, z^{j-1} \\ f_{;j} &= f_0 + \cdots + f_{j-1}\, z^{j-1} \\ f_{i;} &= f_i\, z^i + f_{i+1}\, z^{i+1} + \cdots. \end{aligned}$$

The fast relaxed algorithms from [21] are all based on two main changes of representation: "blocking" and the fast Fourier transform. Let us briefly recall these transformations and how to use them for the design of fast algorithms for relaxed multiplication.

Blocking and unblocking. Given a block size $b > 0$, the first operation of *blocking* rewrites a power series $f \in \mathbb{A}[[z]]$ as a series in $y = z^b$ with coefficients in $\mathbb{A}[z]_b$

$$\mathrm{B}_b(f) \;=\; \sum_i \sum_{0 \leqslant j < b} f_{ib+j}\, z^j\, y^i \;\in\; \mathbb{A}[z]_b[[y]].$$

Given $f, g \in \mathbb{A}[[z]]$, we may then compute $f\,g$ using

$$f\,g \;=\; \mathrm{B}_b^{-1}(\mathrm{B}_b(f)\,\mathrm{B}_b(g)),$$

where $\mathrm{B}_b(f)\,\mathrm{B}_b(g) \in \mathbb{A}[z]_{2b}[[y]]$ and

$$\begin{aligned} \mathrm{B}_b^{-1} : \mathbb{A}[z]_{2b}[[y]] &\to \mathbb{A}[[z]] \\ \sum_i P_i(z)\, y^i &\mapsto \sum_i P_i(z)\, z^{bi}. \end{aligned}$$

Discrete Fourier transforms. Assume now that $\mathbb{A} \ni 1/2$, that $b \in \{1, 2, 4, 8, \ldots\}$, and that $\mathbb{A}$ admits a primitive $(2\,b)$-th root of unity $\omega = \omega_{2b}$. Then the discrete Fourier transform provides us with an isomorphism

$$\begin{aligned} \mathrm{FFT}_\omega : \mathbb{A}[z]_{2b} &\to \mathbb{A}^{2b} \\ P &\mapsto (P(1), P(\omega), \ldots, P(\omega^{2b-1})), \end{aligned}$$

and it is classical [12] that both FFT_ω and FFT_ω^{-1} can be computed using $\mathcal{O}(b \log b)$ operations in $\mathbb{A}$. The operations FFT_ω and FFT_ω^{-1} extend naturally to $\mathbb{A}[z]_{2b}[[y]]$ *via*

$$\mathrm{FFT}_\omega\Big(\sum_i f_i\, y^i\Big) \;=\; \sum_i \mathrm{FFT}_\omega(f_i)\, y^i.$$

Given $f, g \in \mathbb{A}[[z]]$, this allows us to compute $f\,g$ using the formula

$$f\,g = \mathrm{B}_b^{-1}(\mathrm{FFT}_\omega^{-1}(\mathrm{FFT}_\omega(\mathrm{B}_b(f))\,\mathrm{FFT}_\omega(\mathrm{B}_b(g)))), \tag{6}$$

where $\mathrm{FFT}_\omega(\mathrm{B}_b(f))\,\mathrm{FFT}_\omega(\mathrm{B}_b(g))$ is a pointwise product in $\mathbb{A}^{2b}$. The first $m\,b$ coefficients of $f\,g$ can be computed using at most $2\,b\,\mathsf{M}_{\mathbb{A}}(m) + \mathcal{O}(m\,b \log b)$ operations in $\mathbb{A}$.

Relaxed multiplication. In formula (6) the m-th coefficient of the right hand side may depend on the $(m + b - 1)$-th coefficients of f and g. In order to make (6) suitable for relaxed multiplication, we have to treat the first b coefficients of f and g separately. Indeed, the formula

$$\begin{aligned} f\,g \;=\;& f_{;b}\,g_{;b} + f_{;b}\,g_{b;} + f_{b;}\,g_{;b} + \\ & \mathrm{B}_b^{-1}(\mathrm{FFT}_\omega^{-1}(\mathrm{FFT}_\omega(\mathrm{B}_b(f_{b;}))\,\mathrm{FFT}_\omega(\mathrm{B}_b(g_{b;})))) \end{aligned}$$

allows for the relaxed computation of $f\,g$ at order $m\,b$ using at most

$$\begin{aligned} \mathsf{R}_{\mathbb{A}}(m\,b) \;\leqslant\;& \mathsf{R}_{\mathbb{A}}(b) + 2\,m\,\mathsf{S}_{\mathbb{A}}(b) + 2\,b\,\mathsf{R}_{\mathbb{A}}(m) + \\ & \mathcal{O}(m\,b \log b) \end{aligned}$$

operations in $\mathbb{A}$. Similarly, the formula

$$\begin{aligned} f\,g \;&=\; f_{;b}\,g + \mathrm{B}_b^{-1}(\mathrm{FFT}_\omega^{-1}(*)) \\ * \;&=\; \mathrm{FFT}_\omega(\mathrm{B}_b(f_{b;}))\,\mathrm{FFT}_\omega(\mathrm{B}_b(g)) \end{aligned} \tag{7}$$

allows for the semi-relaxed computation of $f\,g$ at order $m\,b$ using at most

$$\mathsf{S}_{\mathbb{A}}(m\,b) \;\leqslant\; m\,\mathsf{S}_{\mathbb{A}}(b) + 2\,b\,\mathsf{S}_{\mathbb{A}}(m) + \mathcal{O}(m\,b \log b) \tag{8}$$

operations in $\mathbb{A}$. For a given expansion order n, one may take $b \approx \sqrt{n}$, and use the above formula in a recursive manner. This yields [21, Theorem 11]

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(n\,(\log n)^{\log 3/\log 2}).$$

Remark 4. Since the block size b is chosen as a function of n, the above method really describes a relaxed algorithm for computing the product up to an order n_* which is specified in advance. In fact, such an algorithm automatically yields a genuine relaxed algorithm with the same complexity (up to a constant factor), by doubling the order n_* each time when needed.

Reduction to semi-relaxed multiplication. In the above discussion, we both provided bounds for $\mathsf{R}_{\mathbb{A}}(n)$ and $\mathsf{S}_{\mathbb{A}}(n)$. In fact, there exists a straightforward reduction of relaxed multiplication to semi-relaxed multiplication. First of all, the relaxed multiplication of two power series $f,\ g \in \mathbb{A}[[z]]$ up to order $\mathcal{O}(z^n)$ clearly reduces to the relaxed multiplication of the two polynomials $f_{;n}$ and $g_{;n}$ up to order $\mathcal{O}(z^{2n})$. Now the formula

$$f_{;2n}\,g_{;2n} \;=\; f_{;n}\,g_{;n} + f_{n;2n}\,g_{;n} + f_{;n}\,g_{n;2n} + f_{n;2n}\,g_{n;2n}$$

shows that a relaxed product of two polynomials $f_{;2n}$ and $g_{;2n}$ of degrees $<(2\,n)$ reduces to a relaxed product $f_{;n}\,g_{;n}$ of half the size, two semi-relaxed products $f_{n;2n}\,g_{;n}$, $f_{;n}\ g_{n;2n}$, and one non-relaxed product $f_{n;2n}\ g_{n;2n}$. Under the assumptions that $\mathsf{M}_{\mathbb{A}}(n)/n$ and $\mathsf{S}_{\mathbb{A}}(n)/n$ are increasing, a routine calculation thus yields

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{S}_{\mathbb{A}}(n)).$$

Multiple block sizes. Instead of using a single block size b, one may use several block sizes. Applying this technique, we proved in [21, Theorem 12] that

$$\begin{aligned} \mathsf{R}_{\mathbb{A}}(n) \;&=\; \mathcal{O}(\mathsf{S}_{\mathbb{A}}(n)) \\ &=\; \mathcal{O}(n \log n\, \mathrm{e}^{2\sqrt{\log 2}\sqrt{\log\log n}}). \end{aligned} \tag{9}$$

4 Improved complexity analysis

One of the referees suggested to take $b \approx n^{2/3}$ instead of $b \approx \sqrt{n}$ in (8). As a matter of fact, it is even better to take $b \approx \exp(\log n / \exp(\sqrt{2 \log 2}\, \sqrt{\log\log n}))$. In this section, we will show that this leads to the bound

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_*(n)),$$

which further improves on (9). We will prove a slightly more general result, which is useful for the analysis of algorithms which satisfy complexity bounds similar to (8).

Lemma 5. *Let $\psi : \mathbb{R}^> \to \mathbb{R}^>$ be an increasing function with $\psi(k) = \mathcal{O}(\log k)$ and let*

$$\Phi(k) \;=\; k\, \mathrm{e}^{a \sqrt{\log k}}\, \sqrt{\log k}\, \psi(k),$$

where $a = \sqrt{2 \log 2}$. Then there exist constants $A > 0$ and k_0 such that for all $k \geqslant k_0$, $\varepsilon_1, \varepsilon_2 \in (-1, 1)$, and $\delta = k / \mathrm{e}^{a \sqrt{\log k}}$, we have

$$\Phi(k - \delta + \varepsilon_1) + 2\, \Phi(\delta + \varepsilon_2) \;\leqslant\; \Phi(k) - A\, k\, \psi(k).$$

Proof. Notice that $k - \delta + \varepsilon_1 \leqslant k$ and $\delta + \varepsilon_2 \leqslant k$ for sufficiently large k. We have

$$\begin{aligned} \mathrm{e}^{a \sqrt{\log (k - \delta + \varepsilon_1)}} &= \mathrm{e}^{a \sqrt{\log k - \mathrm{e}^{-a \sqrt{\log k}} (1 + o(1))}} \\ &= \mathrm{e}^{a \sqrt{\log k}} \left(1 - \frac{a + \mathcal{O}(1)}{2 \sqrt{\log k}\, \mathrm{e}^{a \sqrt{\log k}}} \right). \end{aligned}$$

For a suitable constant $A > 0$ and all sufficiently large k, it follows that

$$\Phi(k - \delta + \varepsilon_1) \leqslant ((k - \delta)\, \mathrm{e}^{a \sqrt{\log k}}\, \sqrt{\log k} - A\, k)\, \psi(k). \tag{10}$$

We also have

$$\begin{aligned} a \sqrt{\log (\delta + \varepsilon_2)} &\leqslant a \sqrt{\log k - a \sqrt{\log k} + \mathcal{O}(1/\log k)} \\ &= a \sqrt{\log k} - \frac{a^2}{2} + \frac{a^3}{8 \sqrt{\log k}} + \mathcal{O}\left(\frac{1}{\log k} \right), \end{aligned}$$

whence

$$\begin{aligned} &\mathrm{e}^{a \sqrt{\log (\delta + \varepsilon_2)}}\, \sqrt{\log (\delta + \varepsilon_2)} \\ &= \frac{\mathrm{e}^{a \sqrt{\log k}}}{2}\, \sqrt{\log k} \left(1 - \frac{4\, a - a^3}{8 \sqrt{\log k}} + \mathcal{O}\left(\frac{1}{\log k} \right) \right) \end{aligned}$$

For all sufficiently large k, it follows that

$$\mathrm{e}^{a \sqrt{\log (\delta + \varepsilon_2)}}\, \sqrt{\log (\delta + \varepsilon_2)} \;\leqslant\; \frac{\mathrm{e}^{a \sqrt{\log k}}}{2}\, \sqrt{\log k}.$$

Consequently,

$$2\, \Phi(\delta + \varepsilon_2) \;\leqslant\; \delta\, \mathrm{e}^{a \sqrt{\log k}}\, \sqrt{\log k}\, \psi(k). \tag{11}$$

Adding up (10) and (11), we obtain

$$\begin{aligned} \Phi(k - \delta + \varepsilon_1) + 2\, \Phi(\delta + \varepsilon_2) &\leqslant (\mathrm{e}^{a \sqrt{\log k}} \log k - A)\, k\, \psi(k) \\ &= \Phi(k) - A\, k\, \psi(k), \end{aligned}$$

for all sufficiently large k. □

Theorem 6. *Let $\mathsf{T}, \phi : \mathbb{N}^* \to \mathbb{R}^>$ be an increasing function with $\phi(n) = \mathcal{O}(\log\log n)$. Assume that*

$$\mathsf{T}(n) \;\leqslant\; m\, \mathsf{T}(b) + 2\, b\, \mathsf{T}(m) + m\, b \log b\, \phi(b), \tag{12}$$

for all sufficiently large n, where

$$\begin{aligned} m &= \lceil \exp(\log n / \mathrm{e}^{a \sqrt{\log\log n}}) \rceil \\ b &= \lceil n / m \rceil. \end{aligned}$$

Then

$$\mathsf{T}(n) \;=\; \mathcal{O}(\mathsf{R}_*(n)\, \phi(n)).$$

Proof. We define functions $\mathsf{U}, \psi : \mathbb{R}^> \to \mathbb{R}^>$ by

$$\begin{aligned} \mathsf{U}(k) &= \frac{\mathsf{T}(\lceil \exp(k) \rceil)}{\exp(k)} \\ \psi(k) &= \phi(\lceil \exp(k) \rceil) \end{aligned}$$

and let Φ, A and k_0 be as in Lemma 5. Without loss of generality, we may pick k_0 sufficiently large such that $k_0 \geqslant \exp(9)$ and such that (12) holds for $n \geqslant n_0 := \lceil \exp(k_0) \rceil$. Let $n \in \mathbb{N}^*$ be such that $n \geqslant n_0$ and denote $k = \log n$ and $\delta = k / \mathrm{e}^{a \sqrt{\log k}}$. For certain $\varepsilon_1, \varepsilon_2 \in (-1, 1)$, we have $\log m = \log \lceil \exp(\log n / \mathrm{e}^{a \sqrt{\log\log n}}) \rceil = \delta + \varepsilon_2$ and $\log b = \log \lceil n / m \rceil = k - \delta + \varepsilon_1$. Now (12) implies

$$\mathsf{U}(k) \;\leqslant\; \mathsf{U}(k - \delta + \varepsilon_1) + 2\, \mathsf{U}(\delta + \varepsilon_2) + 2\, k\, \psi(k).$$

Let $\mathcal{K} = \log \mathbb{N}^*$ and $C = \max \{ \sup_{k \in \mathcal{K}, k \leqslant k_0} \mathsf{U}(k) / \Phi(k),\ 2 / A \}$. Let us prove by induction that $\mathsf{U}(k) \leqslant C\, \Phi(k)$ for all $k \in \mathcal{K}$. This is clear for $k \leqslant k_0$. Assume now that $k \geqslant k_0$ and $\mathsf{U}(k') \leqslant C\, \Phi(k')$ for all $k' < k$. Then, with the above notations, $k_0 \geqslant \exp(9)$ implies $k - \delta + \varepsilon_1 < k$ and $\delta + \varepsilon_2 < k$, whence

$$\begin{aligned} \mathsf{U}(k) &\leqslant \mathsf{U}(k - \delta + \varepsilon_1) + 2\, \mathsf{U}(\delta + \varepsilon_2) + 2\, k\, \psi(k), \\ &\leqslant C\, (\Phi(k - \delta + \varepsilon_1) + 2\, \Phi(\delta + \varepsilon_2)) + 2\, k\, \psi(k) \\ &\leqslant C\, \Phi(k) + (2 - C\, A)\, k\, \psi(k) \\ &\leqslant C\, \Phi(k), \end{aligned}$$

as desired. For all $n \in \mathbb{N}^*$, we have thus shown that $\mathsf{T}(n) \leqslant C\, n\, \Phi(\log n) = C\, \mathsf{R}_*(n)\, \phi(n)$. □

5 Relaxed multiplication in characteristic zero

Let us now consider the less favourable case when $\mathbb{A}$ is an effective ring which does not necessarily contain primitive 2^k-th roots of unity for arbitrarily high k. In this section, we will first consider the case when $\mathbb{A}$ is torsion-free as a $\mathbb{Z}$-module and also admits a partial algorithm for division by integers.

Given a block size $b \in \mathbb{N}$ and $q \in \mathbb{Z} \setminus \{-1, 0, 1\}$ (say $q = 2$), we will replace the discrete Fourier transform FFT_ω at a $(2\, b)$-th primitive root of unity by multipoint evaluation at $1, \ldots, q^{2b-1}$. More precisely, we define

$$\begin{aligned} \mathrm{E}_{q,2b} : \mathbb{A}[z]_{2b} &\to \mathbb{A}^{2b} \\ P &\mapsto (P(1), P(q), \ldots, P(q^{2b-1})) \end{aligned}$$

and the inverse transform $\mathrm{E}_{q,2b}^{-1} : \operatorname{im} \mathrm{E}_{q,2b} \to \mathbb{A}[z]_{2b}$. By Lemma 3, these transforms can both be computed using $\mathcal{O}(\mathsf{M}_{\mathbb{A}}(b))$ operations in $\mathbb{A}$. In a similar way as for FFT_ω and FFT_ω^{-1}, we extend $\mathrm{E}_{q,2b}$ and $\mathrm{E}_{q,2b}^{-1}$ to power series in y.

Theorem 7. *Let $\mathbb{A}$ both be an effective ring and an effective torsion-free $\mathbb{Z}$-module with an effective partial division by elements in $\mathbb{Z} \setminus \{0\}$. Then*

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_{**}(n)).$$

Proof. It suffices to prove the complexity bound for semi-relaxed multiplication. Instead of (7), we now compute $f\, g$ using

$$\begin{aligned} f\, g &= f_{;b}\, g + \mathrm{B}_b^{-1}(\mathrm{E}_{q,2b}^{-1}(*)), \\ * &= \mathrm{E}_{q,2b}(\mathrm{B}_b(f_{b;}))\, \mathrm{E}_{q,2b}(\mathrm{B}_b(g)). \end{aligned} \tag{13}$$

The bound (8) then has to be replaced by

$$\mathsf{S}_{\mathbb{A}}(m\,b) \;\leqslant\; m\,\mathsf{S}_{\mathbb{A}}(b) + 2\,b\,\mathsf{S}_{\mathbb{A}}(m) + \mathcal{O}(m\,\mathsf{M}_{\mathbb{A}}(b)). \tag{14}$$

Plugging in the bound $\mathsf{M}_{\mathbb{A}}(b) = \mathcal{O}(b \log b \log\log b)$ from [10] and applying Theorem 6, the result follows. □

6 Relaxed multiplication in prime characteristic

Let $\mathbb{A}$ now be an effective ring of prime characteristic p. For expansion orders $n \geqslant p$, the ring $\mathbb{A}$ does not necessarily contain n distinct points in geometric progression. Therefore, in order to apply Lemma 3, we will first replace $\mathbb{A}$ by a suitable extension, in which we can find sufficiently large geometric progressions.

Given n, let $k = 2\left\lceil \frac{\log(n+1)}{2\log p} \right\rceil$ be even with $p^k > n$. Let $P \in \mathbb{F}_p[z]$ be such that the finite field $\mathbb{F}_{p^k}$ is isomorphic to $\mathbb{F}_p[z]/(P)$. Then the ring

$$\mathbb{B} \;:=\; \mathbb{A}[z]/(P)$$

has dimension k over $\mathbb{A}$ as a vector space, so we have a natural $\mathbb{A}$-linear bijection

$$\begin{aligned} \Lambda\colon \mathbb{A}[z]_k &\to \mathbb{B} \\ A &\mapsto A \bmod P. \end{aligned}$$

The ring $\mathbb{B}$ is an effective ring and one addition or subtraction in $\mathbb{B}$ corresponds to k additions or subtractions in $\mathbb{A}$. Similarly, one multiplication in $\mathbb{B}$ can be done using $\mathcal{O}(\mathsf{M}_{\mathbb{A}}(k))$ operations in $\mathbb{A}$.

In order to multiply two series $f, g \in \mathbb{A}[[z]]$ up to order $\mathcal{O}(z^n)$, the idea is now to rewrite f and g as series in $\mathbb{B}[[u]]$ with $u = z^{k/2}$. If we want to compute the relaxed product, then we also have to treat the first $k/2$ coefficients apart, as we did before for the blocking strategy. More precisely, we will compute the semi-relaxed product $f\,g$ using the formula

$$\begin{aligned} f\,g &= f_{;k/2}\,g + \mathrm{B}_{k/2}^{-1}(\Lambda^{-1}(*)), \\ * &= \Lambda(\mathrm{B}_{k/2}(f_{k/2;}))\,\Lambda(\mathrm{B}_{k/2}(g)), \end{aligned}$$

where we extended Λ to $\mathbb{A}[z]_k[[u]]$ in the natural way:

$$\Lambda\left(\sum_{i \geqslant 0} f_i\,u^i\right) \;=\; \sum_{i \geqslant 0} \Lambda(f_i)\,u^i.$$

From the complexity point of view, we get

$$\mathsf{S}_{\mathbb{A}}(n) \;\leqslant\; \frac{2\,n}{k}\,\mathsf{S}_{\mathbb{A}}\left(\frac{k}{2}\right) + \mathsf{S}_{\mathbb{B}}\left(\frac{2\,n}{k}\right)\mathcal{O}(\mathsf{M}_{\mathbb{A}}(k)). \tag{15}$$

Since $\mathbb{B}$ contains a copy of $\mathbb{F}_{p^k}$, it also contains at least $p^k - 1 \geqslant n$ points in geometric progression. For the multiplication up to order $2\,n/k$ of two series with coefficients in $\mathbb{B}$, we may thus use the blocking strategy combined with multipoint evaluation and interpolation.

Theorem 8. *Let $\mathbb{A}$ be an effective ring of prime characteristic p. Then*

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_{***}(n)).$$

Proof. With the notations from above, we may find a primitive $(p^k - 1)$-th root of unity q in $\mathbb{F}_{p^k} \subseteq \mathbb{B}$. We may thus use formula (13) for the semi-relaxed multiplication of two series in $\mathbb{B}[[z]]$ up to order $2\,n/k \leqslant n$. In a similar way as in the proof of Theorem 7, we thus get

$$\mathsf{S}_{\mathbb{B}}\left(\frac{2\,n}{k}\right) = \mathcal{O}\left(\mathsf{R}_{**}\left(\frac{2\,n}{k}\right)\right) = \mathcal{O}\left(\frac{\mathsf{R}_{**}(n)}{k}\right).$$

Using classical fast relaxed multiplication [16, 17, 13], we also have

$$\mathsf{S}_{\mathbb{A}}\left(\frac{k}{2}\right) \;=\; \mathcal{O}(k \log^2 k \log\log k),$$

whence (15) simplifies to

$$\mathsf{S}_{\mathbb{A}}(n) \;=\; \mathcal{O}\left(\mathsf{R}_{**}(n)\,\frac{\mathsf{M}_{\mathbb{A}}(k)}{k}\right). \tag{16}$$

Since $\mathsf{M}_{\mathbb{A}}(k)/k = \mathcal{O}(\log k \log\log k)$ and $k = \mathcal{O}(\log n)$, the result follows. □

Remark 9. As long as $\log n = \mathcal{O}(\log p)$, then $\mathsf{M}_{\mathbb{A}}(k)/k = \mathcal{O}(1)$ in (16), so the bound further reduces into

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_{**}(n)).$$

Remark 10. In our complexity analysis, we have not taken into account the computation of the polynomial $P \in \mathbb{F}_p[z]$ with $\mathbb{F}_{p^k} = \mathbb{F}_p[z]/(P)$. Using a randomized algorithm, such a polynomial can be computed in time $\tilde{\mathcal{O}}(k^2 \log p)$; see [15, Corollary 14.44]. If $k = \mathcal{O}(\log n)$, then this is really a precomputation of negligible cost $\tilde{\mathcal{O}}(\log^2 n \log p)$.

If we insist on computing P in a deterministic way, then one may use [30, Theorem 3.2], which provides us with an algorithm of time complexity $\tilde{O}(\sqrt{p}\,k^{4+\varepsilon})$.

Similarly, there both exist randomized and deterministic algorithms [32, 31] for the efficient computation of primitive $(p^k - 1)$-th roots of unity in $\mathbb{F}_{p^k}$. In particular, thanks to [31, Theorem 1] such a primitive root of unity can always be computed in time $\tilde{O}(p^{k/2})$, when using a naive algorithm for factoring $p^k - 1$.

7 Relaxed multiplication in positive characteristic

Let us now show that the technique from the previous section actually extends to the case when $\mathbb{A}$ is an arbitrary effective ring of positive characteristic. We first show that the algorithm still applies when the characteristic of $\mathbb{A}$ is a prime power. We then conclude by showing how to apply Chinese remaindering in our setting.

Theorem 11. *Let $\mathbb{A}$ be an effective ring of prime power characteristic $s = p^r$. Then*

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_{***}(n)).$$

Proof. Taking $k = 2\left\lceil \frac{\log(n+1)}{2\log p} \right\rceil$, let P of degree k be as in the previous section and pick a monic polynomial $\tilde{P}$ of degree k in $(\mathbb{Z}/s\,\mathbb{Z})[z]$ such that the reduction $\pi(\tilde{P})$ of $\tilde{P}$ modulo p yields P. Then we get a natural commutative diagram

$$\begin{array}{ccc} (\mathbb{Z}/s\,\mathbb{Z})[z]/(\tilde{P}) & \hookrightarrow & \mathbb{A}[z]/(\tilde{P}) \\ \downarrow\pi & & \downarrow\pi \\ \mathbb{F}_p[z]/(P) & \hookrightarrow & (\mathbb{A}/p\,\mathbb{A})[z]/(P), \end{array}$$

where π stands for reduction modulo p. In particular, we have an epimorphism

$$\pi\colon (\mathbb{Z}/s\,\mathbb{Z})[z]/(\tilde{P}) \;\to\; \mathbb{F}_p[z]/(P) \cong \mathbb{F}_{p^k},$$

with $\ker\pi = (p)$.

Now let q be an element in $\mathbb{F}_{p^k}$ of order p^k-1. Then any lift $\tilde{q} \in (\mathbb{Z}/s\,\mathbb{Z})[z]/\big(\tilde{P}\big)$ of q with $\pi(\tilde{q}) = q$ has order at least p^k-1. Moreover, $q-1$, ..., $q^{p^k-2}-1$ and q are all invertible. Consequently, $\tilde{q}-1$, ..., $\tilde{q}^{p^k-2}-1$ and $\tilde{q}$ do not lie in $\ker \pi = (p)$, whence they are invertible as well. It follows that we may still apply multipoint evaluation and interpolation in $\mathbb{A}[z]/(\tilde{P})$ at the sequence $1, \tilde{q}, ..., \tilde{q}^{p^k-2}$, whence Theorem 8 generalizes to the present case. □

Remark 12. For a fixed prime number p, we notice that the complexity bound is uniform in the following sense: there exists a constant K such that for all effective rings of characteristic p^r with $r \in \{1, 2, ...\}$, we have $\mathsf{R}_{\mathbb{A}}(n) \leqslant K\,\mathsf{R}_{***}(n)$. Indeed, the choice of k only depends on n and p, and any operation in $\mathbb{F}_{p^k}$ or $\mathbb{A}[z]/(P)$ in the case $r = 1$ corresponds to exactly one lifted operation in $(\mathbb{Z}/s\,\mathbb{Z})[z]/(\tilde{P})$ or $\mathbb{A}[z]/(\tilde{P})$ in the general case.

Remark 13. Similarly as in Remark 9, the hypothesis $\log n = \mathcal{O}(\log p)$ leads to the improved bound

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_{**}(n)).$$

This bound is uniform in a similar way as in Remark 12.

Theorem 14. *Let $\mathbb{A}$ be an effective ring of non-zero characteristic s. Then*

$$\mathsf{R}_{\mathbb{A}}(n) \;=\; \mathcal{O}(\mathsf{R}_{***}(n)).$$

Proof. We will prove the theorem by induction on the number of prime divisors of s. If s is a prime power, then we are done. So assume that $s = s_1\,s_2$, where s_1 and s_2 are relatively prime, and let $k_1, k_2 \in \mathbb{Z}$ be such that

$$k_1\,s_1 + k_2\,s_2 \;=\; 1.$$

Then we may consider the rings

$$\begin{aligned} \mathbb{A}_1 &= \mathbb{A}/s_1\,\mathbb{A} \\ \mathbb{A}_2 &= \mathbb{A}/s_2\,\mathbb{A}. \end{aligned}$$

These rings are effective, when representing their elements by elements of $\mathbb{A}$ and transporting the operations from $\mathbb{A}$. Of course, the representation of an element x of $\mathbb{A}_1$ (or $\mathbb{A}_2$) is not unique, since we may replace it by $x + y$ for any $y \in s_1\,\mathbb{A}$ (or $y \in s_2\,\mathbb{A}$). But this is not a problem, since our definition of effective ring did not require unique representability or the existence of an equality test.

Now let $f, g \in \mathbb{A}[[z]]$ and let $\pi_i(f), \pi_i(g)$ be their projections in $\mathbb{A}_i[[z]]$, for $i = 1, 2$. Consider the relaxed products $\pi_i(f)\;\pi_i(g)$, for $i = 1, 2$. These products are represented by relaxed series $h^1, h^2 \in \mathbb{A}[[z]]$ *via* $\pi_i(h^i) = \pi_i(f)\,\pi_i(g)$, for $i = 1, 2$. By the induction hypotheses, we may compute h^1 and h^2 at order n using $\mathcal{O}(\mathsf{R}_{***}(n))$ operations in $\mathbb{A}$. The linear combination $h = k_2\,s_2\,h^1 + k_1\,s_1\,h^2 \in \mathbb{A}[[z]]$ can still be expanded up to order n with the same complexity. We claim that $h = f\,g$. Indeed,

$$\begin{aligned} k_2\,s_2\,h^1 - k_2\,s_2\,f\,g &\in k_2\,s_2\,s_1\,\mathbb{A} = \{0\} \\ k_1\,s_1\,h^2 - k_1\,s_1\,f\,g &\in k_1\,s_1\,s_2\,\mathbb{A} = \{0\}. \end{aligned}$$

Summing both relations, our claim follows. □

Remark 15. A uniform bound interms of s can be given along similar lines as in Remark 12. This time, such a bound depends linearly on the number of prime factors of s.

8 Relaxed multiplication of p-adic numbers

Let $p > 1$ be an integer, not necessarily a prime number, and denote $\mathbb{N}_p = \{0, ..., p-1\}$. We will regard p-adic numbers $a \in \mathbb{Z}_p$ as series $a_0 + a_1\,p + a_2\,p^2 + \cdots$ with $a_i \in \mathbb{N}_p$, and such that the basic ring operations $+$, $-$ and $\times$ require an additional carry treatment.

In order to multiply two relaxed p-adic numbers a, $b \in \mathbb{Z}_p$, we may rewrite them as series $\hat{a}, \hat{b} \in \mathbb{Z}[[z]]$, multiply these series $\hat{c} = \hat{a}\,\hat{b}$, and recover the product $c \in \mathbb{Z}_p$ from the result. Of course, the coefficients of $\hat{c}$ may exceed p, so some relaxed carry handling is required in order to recover c from $\hat{c}$. We refer to [4, Section 2.7] for details. In particular, we prove there that c can be computed up to order $\mathcal{O}(p^n)$ using $\mathcal{O}(\mathsf{R}_{\mathbb{Z}}(n))$ ring operations in $\mathbb{Z}$ of bit size $\mathcal{O}(\log p + \log n)$.

Given $k > 0$, let $\mathcal{Z}_k = \{i \in \mathbb{Z}\colon |i| < 2^{k-1}\}$, and consider two power series $f, g \in \mathcal{Z}_k[[z]]$. We will denote by $\mathsf{R}_{\mathbb{Z}}(n, k)$ (resp. $\mathsf{S}_{\mathbb{Z}}(n, k)$) the bit complexity of multiplying f and g up to order $\mathcal{O}(z^n)$ using a relaxed (resp. semi-relaxed) algorithm.

Lemma 16. *We have*

$$\mathsf{R}_{\mathbb{Z}}(n, k) = \mathcal{O}(\mathsf{R}_{**}(n)\,\mathsf{I}(k + \log n)).$$

Proof. Let π be a prime number with $\log \pi \asymp \log n$. Such a prime π can be computed in time $\mathcal{O}(n)$ using the polynomial time polynomial primality test from [1], together with the Bertrand-Chebychev theorem.

Let $f, g \in \mathcal{Z}_k[[z]]$ and consider $r = \lceil 2\,k \log(2\,n)/\log \pi \rceil$ such that $n\,2^{2k} \leqslant (2\,n)^{2k} < \pi^r$. Let $\hat{f}, \hat{g} \in (\mathbb{Z}/\pi^r\,\mathbb{Z})[[z]]$ be the reductions of f, g modulo π^r. Then $f\,g$ may be reconstructed up to order $\mathcal{O}(z^n)$ from the product $\hat{f}\,\hat{g}$. We thus get

$$\mathsf{R}_{\mathbb{Z}}(n, k) \;=\; \mathcal{O}(\mathsf{I}(\log_2(\pi^r))\,\mathsf{R}_{\mathbb{Z}/\pi^r\mathbb{Z}}(n)).$$

By Theorem 11 and Remarks 12 and 13, while using the fact that $\log n = \mathcal{O}(\log \pi)$, we have

$$\mathsf{R}_{\mathbb{Z}/\pi^r\mathbb{Z}}(n) \;=\; \mathcal{O}(\mathsf{R}_{**}(n)),$$

and this bound is uniform in r. Since $\pi^r \sim n\,2^{2k}$, the result follows. □

For the above strategy to be efficient, it is important that $\log\,n = \mathcal{O}(k)$. This can be achieved by combining it with the technique of p-adic blocking. More precisely, given a p-adic block size $b > 1$, then any p-adic number in $\mathbb{Z}_p$ can naturally be considered as a p^b-adic number in $\mathbb{Z}_{p^b}$, and *vice versa*. Assuming that numbers in $\mathbb{N}_p$ are written in base 2, the conversion is trivial if p is a power of two. Otherwise, the conversion involves base conversions and we refer to [4, Section 4] for more details. In particular, the conversions in both directions up to order $\mathcal{O}(p^n)$ can be done in time $\mathcal{O}\big(\frac{n}{b}\,\mathsf{I}(b \log p) \log(b \log p)\big)$.

Let $\mathsf{R}_p(n)$ (resp. $\mathsf{S}_p(n)$) the complexity of relaxed (resp. semi-relaxed) multiplication in $\mathbb{Z}_p$ up to order $\mathcal{O}(p^n)$.

Theorem 17. *Setting $\ell = \log(\log n + \log p)$, we have*

$$\begin{aligned} \mathsf{R}_p(n) &= \mathcal{O}((\mathsf{R}_{**}(n) \log p)\,\ell \log \ell) \\ &= \mathcal{O}^{\flat}(n \log n \log p \log\log p). \end{aligned}$$

Proof. Let $r = \lceil \log n / \log p \rceil$, so that

$$\begin{aligned} r \log p &\leqslant \log n + \log p \\ \mathsf{I}(r \log p) &= \mathcal{O}((r \log p)\,\ell \log \ell) \\ \mathsf{I}(r\,(\log p + \log r)) &= \mathcal{O}(r\,(\log p + \log n)\,\ell \log \ell). \end{aligned}$$

Using the strategy of p-adic blocking, a semi-relaxed product in $\mathbb{Z}_p$ may then be reduced to one semi-relaxed product in $\mathbb{Z}_{p^r}$ and one relaxed multiplication with an integer in $\{0, ..., p^r - 1\}$. In other words,

$$\mathsf{S}_p(n) \leqslant \frac{n}{r}\mathsf{S}_p(r) + \mathsf{S}_{p^r}\Big(\Big\lceil \frac{n}{r} \Big\rceil\Big) + \mathcal{O}(n \log p),$$

where $\mathsf{S}_p(n)$ stands for the cost of semi-relaxed multiplication of two p-adic numbers in $\mathbb{Z}_p$ up to order $\mathcal{O}(p^n)$. By [4, Proposition 4], we have

$$\begin{aligned} \mathsf{S}_p(r) &= \mathcal{O}(\mathsf{I}(r\,(\log p + \log r))\log r) \\ &= \mathcal{O}(r \log n\,(\log n + \log p)\,\ell\log\ell). \end{aligned}$$

By Lemma 16, we also have

$$\begin{aligned} \mathsf{S}_{p^r}\big(\lceil \tfrac{n}{r} \rceil\big) &= \mathcal{O}\big(\mathsf{R}_{\mathbb{Z}}\big(\tfrac{n}{r}, \log_2(p^r)\big)\big) \\ &= \mathcal{O}\big(\mathsf{I}(r\log p + \log n)\,\tfrac{n}{r}\log n \\ &\qquad \mathrm{e}^{\sqrt{2\log 2}\sqrt{\log\log n}}\,(\log\log n)^{3/2}\big) \\ &= \mathcal{O}\big(\mathsf{I}(r\log p)\,\tfrac{1}{r}\mathsf{R}_{**}(n)\big) \\ &= \mathcal{O}((\log p)\,\ell\log\ell\,\mathsf{R}_{**}(n)) \end{aligned}$$

Notice that

$$\frac{n}{r}\mathsf{S}_p(r) = \mathcal{O}\Big(\mathsf{S}_{p^r}\Big(\Big\lceil \frac{n}{r} \Big\rceil\Big)\Big),$$

which completes the proof of the theorem. □

9 Final remarks

For the moment, we have not implemented any of the new algorithms in practice. Nevertheless, our old implementation of the algorithm from [21] allowed us to gain some insight on the practical usefulness of blockwise relaxed multiplication. Let us briefly discuss the potential impact of the new results for practical purposes.

Characteristic zero. In characteristic zero, our focus on algebraic complexity makes the complexity bounds more or less irrelevant from a practical point of view. In practice, two cases are of particular interest: floating point coefficients (which were already considered in [21]) and integer coefficients (rational coefficients can be dealt with similarly after multiplying by the common denominator).

In the case of integer coefficients, it is best to re-encode the integers as polynomials in $\mathbb{F}_p[x]$ for a prime number which fits into a machine word and such that $\mathbb{F}_p$ admits many 2^k-th roots of unity (it is also possible to take several primes p and use Chinese remaindering). After that, one may again use the old algorithm from [21]. Also, integer coefficients usually grow in size with n, so one really should see the power series as a bivariate power series in $\mathbb{F}_p[[x, z]]$ with a triangular support. One may then want to use TFT-style multiplication [19, 20] in order to gain another constant factor.

Finite fields. For large finite fields, it is easy to find large geometric progressions, so the algorithms of this paper can be applied without the need to consider field extensions. Moreover, for finite fields of the form $\mathbb{F}_{p^k}$ with p sufficiently large and $k > 1$, it is possible to choose $q \in \mathbb{F}_p$, thereby speeding up evaluation and interpolation. For small finite fields of the form $\mathbb{F}_{p^k}$, it is generally necessary to make the *initial investment* of working in a larger ring with sufficiently large geometric progressions. Of course, instead of the ring extensions considered in Section 6, we may directly use field extensions of the form $\mathbb{F}_{p^l}$ with $k \,|\, l$.

Semi-relaxed multiplication. In principle, a factor 2 can be gained in the semi-relaxed (resp. general) case using the technique from [18] (resp. [26]). Unfortunately, the middle product (resp. the FFT-trick used in [26]) is not always easy to implement. For instance, if we rely on Kronecker substitution for multiplications in $\mathbb{Z}[z]$, then we will need to implement an *ad hoc* analogue for the middle product. Since we did not use a fast algorithm for middle products for our benchmarks in [21, Section 5], the timings for the semi-relaxed product in were only about 25% instead of 50% better than the timings for the fully relaxed product. Nevertheless, modulo increased implementation efforts, we stress that a 50% gain should be achievable.

Cache friendliness. So far, we have not investigated the cache friendliness of blockwise relaxed multiplication, and it can be feared that a lot of additional work is required in order to make our algorithms really efficient from this point of view.

Bilinear maps. In order to keep the presentation reasonably simple, we have focussed on the case when $\mathbb{A}$ is an effective ring. In fact, a more general setting for relaxed multiplication is to consider a bilinear mapping $\mu\colon \mathbb{M}_1 \times \mathbb{M}_2 \to \mathbb{M}_3$, where $\mathbb{M}_1$, $\mathbb{M}_2$ and $\mathbb{M}_3$ are effective $\mathbb{A}$-modules, and extend it into a mapping $\hat{\mu}\colon \mathbb{M}_1[[z]] \times \mathbb{M}_2[[z]] \to \mathbb{M}_3[[z]]$ by $\hat{\mu}(f, g) = \sum_{i,j} \mu(f_i, g_j)\, z^{i+j}$. Under suitable hypothesis, the algorithms in this paper generalize to this setting.

Skew series. The relaxed approach can also be generalized to the case when the coefficients of the power series are operators which commute with monomials z^i in a non-trivial way. More precisely, assume that we have an effective ring homomorphism $\phi\colon \mathbb{A} \to \mathbb{A}$ such that $z\,a = (\phi a)\,z$ for all $a \in \mathbb{K}$. For instance, one may take $\mathbb{A} = \mathbb{Q}[\delta]$ with $\delta = z\,\partial/\partial z$, so that $P(\delta)\,z = z\,P(\delta + 1)$. Given a commutation rule of this kind, we define a skew multiplication on $\mathbb{A}[[z]]$ by

$$\left[\sum_{i \geqslant 0} f_i\, z^i\right]\left[\sum_{j \geqslant 0} g_j\, z^j\right] = \sum_{i,j \geqslant 0} f_i\,(\phi^i\, g_j)\, z^{i+j}.$$

If $\mathsf{M}_{\mathbb{A}}(n)$ denotes the cost of multiplying two polynomials $P\,z^i$ and $Q\,z^j$ in $\mathbb{A}[z]$ with $\deg P$, $\deg Q < n$, then the classical fast relaxed multiplication algorithm from [17, 13] generalizes and still admits the time complexity $\mathcal{O}(\mathsf{M}_{\mathbb{A}}(n)\log n)$. However, the blockwise algorithm from this paper does not generalize to this setting, at least not in a straightforward way.

Bibliography

[1] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004.

[2] A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, Massachusetts, 1974.

[3] D. Bernstein. Removing redundancy in high precision Newton iteration. Available from `http://cr.yp.to/fastnewton.html`, 2000.

[4] J. Berthomieu, J. van der Hoeven, and G. Lecerf. Relaxed algorithms for p-adic numbers. *Journal de Théorie des Nombres de Bordeaux*, 23(3):541–577, 2011.

[5] J. Berthomieu and R. Lebreton. Relaxed p-adic hensel lifting for algebraic systems. In J. van der Hoeven and M. van Hoeij, editors, *Proc. ISSAC '12*, pages 59–66, Grenoble, France, July 2012.

[6] D. Bini and V.Y. Pan. *Polynomial and matrix computations. Vol. 1*. Birkhäuser Boston Inc., Boston, MA, 1994. Fundamental algorithms.

[7] Leo I. Bluestein. A linear filtering approach to the computation of discrete Fourier transform. *IEEE Transactions on Audio and Electroacoustics*, 18(4):451–455, 1970.

[8] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *Journal of Complexity*, 21(4):420–446, August 2005. Festschrift for the 70th Birthday of Arnold Schönhage.

[9] R.P. Brent and H.T. Kung. Fast algorithms for manipulating formal power series. *Journal of the ACM*, 25:581–595, 1978.

[10] D.G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28:693–701, 1991.

[11] S. A. Cook. *On the minimum computation time of functions*. PhD thesis, Harvard University, 1966.

[12] J.W. Cooley and J.W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Computat.*, 19:297–301, 1965.

[13] M. J. Fischer and L. J. Stockmeyer. Fast on-line integer multiplication. *Proc. 5th ACM Symposium on Theory of Computing*, 9:67–72, 1974.

[14] M. Fürer. Faster integer multiplication. In *Proceedings of the Thirty-Ninth ACM Symposium on Theory of Computing (STOC 2007)*, pages 57–66, San Diego, California, 2007.

[15] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, first edition, 1999.

[16] J. van der Hoeven. Lazy multiplication of formal power series. In W. W. Küchlin, editor, *Proc. ISSAC '97*, pages 17–20, Maui, Hawaii, July 1997.

[17] J. van der Hoeven. Relax, but don't be too lazy. *JSC*, 34:479–542, 2002.

[18] J. van der Hoeven. Relaxed multiplication using the middle product. In Manuel Bronstein, editor, *Proc. ISSAC '03*, pages 143–147, Philadelphia, USA, August 2003.

[19] J. van der Hoeven. The truncated Fourier transform and applications. In J. Gutierrez, editor, *Proc. ISSAC 2004*, pages 290–296, Univ. of Cantabria, Santander, Spain, July 4–7 2004.

[20] J. van der Hoeven. Notes on the Truncated Fourier Transform. Technical Report 2005-5, Université Paris-Sud, Orsay, France, 2005.

[21] J. van der Hoeven. New algorithms for relaxed multiplication. *JSC*, 42(8):792–802, 2007.

[22] J. van der Hoeven. Relaxed resolution of implicit equations. Technical report, HAL, 2009. http://hal.archives-ouvertes.fr/hal-00441977.

[23] J. van der Hoeven. Newton's method and FFT trading. *JSC*, 45(8):857–878, 2010.

[24] J. van der Hoeven. From implicit to recursive equations. Technical report, HAL, 2011. http://hal.archives-ouvertes.fr/hal-00583125.

[25] A. Karatsuba and J. Ofman. Multiplication of multidigit numbers on automata. *Soviet Physics Doklady*, 7:595–596, 1963.

[26] R. Lebreton and É. Schost. A simple and fast online power series multiplication and its analysis. Technical report, LIRMM, 2013. http://hal-lirmm.ccsd.cnrs.fr/lirmm-00867279.

[27] L.R. Rabiner, R.W. Schafer, and C.M. Rader. The chirp z-transform algorithm and its application. *Bell System Tech. J.*, 48:1249–1292, 1969.

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

[29] Alexandre Sedoglavic. *Méthodes seminumériques en algèbre différentielle ; applications à l'étude des propriétés structurelles de systèmes différentiels algébriques en automatique*. PhD thesis, École polytechnique, 2001.

[30] V. Shoup. New algorithms for finding irreducible polynomials over finite fields. *Math. Comp.*, 54:535–447, 1990.

[31] V. Shoup. Searching for primitive roots in finite fields. *Math. Comp.*, 58:369–380, 1992.

[32] I. Shparlinski. On primitive elements in finite fields and on elliptic curves. *Mat. Sbornik*, 181(9):1196–1206, 1990.

[33] A.L. Toom. The complexity of a scheme of functional elements realizing the multiplication of integers. *Soviet Mathematics*, 4(2):714–716, 1963.