Polynomial multiplication over finite fields in time O(n log n)

David Harvey, Joris van Der Hoeven

Polynomial multiplication over finite fields in time $O(n \log n)$

DAVID HARVEY
School of Mathematics and Statistics
University of New South Wales
Sydney NSW 2052
Australia
Email: d.harvey@unsw.edu.au

JORIS VAN DER HOEVEN
CNRS, Laboratoire d'informatique
École polytechnique
91128 Palaiseau Cedex
France
Email: vdhoeven@lix.polytechnique.fr

March 18, 2019

Assuming a widely-believed hypothesis concerning the least prime in an arithmetic progression, we show that two n-bit integers can be multiplied in time $O(n \log n)$ on a Turing machine with a finite number of tapes; we also show that polynomials of degree less than n over a finite field $\mathbb{F}_q$ with q elements can be multiplied in time $O(n \log q \log(n \log q))$, uniformly in q.

KEYWORDS: Integer multiplication, polynomial multiplication, algorithm, complexity bound, FFT, finite field

A.C.M. SUBJECT CLASSIFICATION: G.1.0 Computer-arithmetic

A.M.S. SUBJECT CLASSIFICATION: 68W30, 68Q17, 68W40

1. INTRODUCTION

Let $\mathsf{I}(n)$ denote the bit complexity of multiplying two integers of at most n bits in the deterministic multitape Turing model [16]. Similarly, let $\mathsf{M}_q(n)$ denote the bit complexity of multiplying two polynomials of degree $<n$ in $\mathbb{F}_q[x]$. Here $\mathbb{F}_q$ is the finite field with q elements, so that $q = \pi^\kappa$ for some prime number π and integer $\kappa > 0$. For computations over $\mathbb{F}_q$, we tacitly assume that we are given a monic irreducible polynomial $\mu \in \mathbb{F}_\pi[x]$ of degree κ, and that elements of $\mathbb{F}_q$ are represented by polynomials in $\mathbb{F}_\pi[x]$ of degree $<\kappa$. Notice that multiplication in $\mathbb{F}_\pi[x]$ can also be regarded as "carryless" integer multiplication in base π.

The best currently known bounds for $\mathsf{I}(n)$ and $\mathsf{M}_q(n)$ were obtained in [21] and [23]. For constants $K_{\mathbb{Z}} = 4$ and $K_{\mathbb{F}} = 4$, we proved there that

$$\mathsf{I}(n) = O\big(n \log n K_{\mathbb{Z}}^{\log^* n}\big) \tag{1.1}$$

$$\mathsf{M}_q(n) = O\big(n \log q \log (n \log q) K_{\mathbb{F}}^{\log^* (n \log q)}\big), \tag{1.2}$$

where $\log x = \ln x$ denotes the natural logarithm of x,

$$\begin{aligned} \log^* x &:= \min \{k \in \mathbb{N} : \log^{\circ k} x \leqslant 1\} \\ \log^{\circ k} &:= \log \circ \underset{k\times}{\cdots} \circ \log, \end{aligned} \tag{1.3}$$

and the bound (1.2) holds uniformly in q.

For the statement of our new results, we need to recall the concept of a "Linnik constant". Given two integers $k > 0$ and $a \in \{1, \ldots, k-1\}$ with $\gcd(a,k) = 1$, we define

$$\begin{aligned} P(a,k) &:= \min \{ck + a : c \in \mathbb{N}, ck + a \text{ is prime}\} \\ P(k) &:= \max \{P(a,k) : 0 < a < k, \gcd(a,k) = 1\}. \end{aligned}$$

We say that a number $L > 1$ is a *Linnik constant* if $P(k) = O(k^L)$. The smallest currently known Linnik constant $L = 5.18$ is due to Xylouris [57]. It is widely believed that any number $L > 1$ is actually a Linnik constant; see section 5 for more details.

In this paper, we prove the following two results:

THEOREM 1.1. *Assume that there exists a Linnik constant with* $L < 1 + \frac{1}{303}$. *Then*

$$\mathsf{I}(n) = O(n \log n).$$

THEOREM 1.2. *Assume that there exists a Linnik constant with* $L < 1 + 2^{-1162}$. *Then*

$$\mathsf{M}_q(n) = O(n \log q \log(n \log q)), \qquad \textit{uniformly in } q.$$

Theorem 1.2 is our main result. Notice that it implies $\mathsf{M}_q(n) = O(b \log b)$ in terms of the bit-size $b = O(n \log q)$ of the inputs. In the companion paper [22], we show that the bound $\mathsf{I}(n) = O(n \log n)$ actually holds unconditionally, thereby superseding Theorem 1.1. Since the proof of Theorem 1.1 is a good introduction to the proof of Theorem 1.2 and also somewhat shorter than the proof of the unconditional bound $\mathsf{I}(n) = O(n \log n)$ in [22], we decided to include it in the present paper. The assumption $L < 1 + 1/303$ being satisfied "in practice", it is also conceivable that variants of the corresponding algorithm may lead to faster practical implementations.

This companion paper [22] uses some of the techniques from the present paper, as well as a new ingredient: Gaussian resampling. This technique makes essential use of properties of real numbers, explaining why we were only able to apply it to integer multiplication. By contrast, the techniques of the present paper can be used in combination with both complex and number theoretic Fourier transforms. For our presentation, we preferred the second option, which is also known to be suitable for practical implementations [17]. It remains an open question whether Theorem 1.2 can be replaced by an unconditional bound.

In the sequel, we focus on the study of $\mathsf{I}(n)$ and $\mathsf{M}_q(n)$ from the purely theoretical perspective of asymptotic complexity. In view of past experience [17, 26, 32], variants of our algorithms might actually be relevant for machine implementations, but these aspects will be left aside for now.

1.1. Brief history and related work

Integer multiplication. The development of efficient methods for integer multiplication can be traced back to the beginning of mathematical history. Multiplication algorithms of complexity $O(n^2)$ were already known in ancient civilizations, whereas descriptions of methods close to the ones that we learn at school appeared in Europe during the late Middle Ages. For historical references, we refer to [54, section II.5] and [41, 6].

The first more efficient method for integer multiplication was discovered in 1962, by Karatsuba [34, 33]. His method is also the first in a family of algorithms that are based on the technique of evaluation-interpolation. The input integers are first cut into pieces, which are taken to be the coefficients of two integer polynomials. These polynomials are evaluated at several well-chosen points. The algorithm next recursively multiplies the obtained integer values at each point. The desired result is finally obtained by pasting together the coefficients of the product polynomial. This way of reducing integer multiplication to polynomial multiplication is also known as Kronecker segmentation [27, section 2.6].

Karatsuba's original algorithm cuts each input integer in two pieces and then uses three evaluation points. This leads to an algorithm of complexity $O(n^{\log 3/\log 2})$. Shortly after the publication of Karatsuba's method, it was shown by Toom [56] that the use of more evaluation points allowed for even better complexity bounds, with further improvements by Schönhage [47] and Knuth [35].

The development of efficient algorithms for the required evaluations and interpolations at large sets of points then became a major bottleneck. Cooley and Tukey's rediscovery [9] of the fast Fourier transform (FFT) provided the technical tool to overcome this problem. The FFT, which was essentially already known to Gauss [29], can be regarded as a particularly efficient evaluation-interpolation method for special sets of evaluation points. Consider a ring R with a principal 2^k-th root of unity $\omega \in R$ (see section 2.2 for detailed definitions; if $R = \mathbb{C}$, then one may take $\omega = \exp(2\pi \mathrm{i}/2^k)$). Then the FFT permits evaluation at the points $1, \omega, \ldots, \omega^{2^k-1}$ using only $O(k2^k)$ ring operations in R. The corresponding interpolations can be done with the same complexity, provided that 2 is invertible in R. Given $P, Q \in R[X]$ with $\deg(PQ) < 2^k$, it follows that the product PQ can be computed using $O(k2^k)$ ring operations as well.

The idea to use fast Fourier transforms for integer multiplication is independently due to Pollard [45] and Schönhage–Strassen [49]. The simplest way to do this is to take $R = \mathbb{C}$ and $\omega = \exp(2\pi \mathrm{i}/2^k)$, while approximating elements of $\mathbb{C}$ with finite precision. Multiplications in $\mathbb{C}$ itself are handled recursively, through reduction to integer multiplications. This method corresponds to Schönhage and Strassen's first algorithm from [49] and they showed that it runs in time $O(n \log n \cdots \log^{\circ(\ell-1)} n \, (\log^{\circ \ell} n)^2)$ for all ℓ. Pollard's algorithm rather works with $R = \mathbb{F}_\pi$, where π is a prime of the form $\pi = a2^k + 1$. The field $\mathbb{F}_\pi$ indeed contains primitive 2^k-th roots of unity and FFTs over such fields are sometimes called number theoretic FFTs. Pollard did not analyze the asymptotic complexity of his method, but it can be shown that its recursive use for the arithmetic in $\mathbb{F}_\pi$ leads to a similar complexity bound as for Schönhage–Strassen's first algorithm.

The paper [49] also contains a second method that achieves the even better complexity bound $\mathsf{I}(n) = O(n \log n \log\log n)$. This algorithm is commonly known as *the* Schönhage–Strassen algorithm. It works over $R = \mathbb{Z}/m\mathbb{Z}$, where $m = 2^{2^k} + 1$ is a Fermat number, and uses $\omega = 2$ as a principal 2^{k+1}-th root of unity in R. The increased speed is due to the fact that multiplications by powers of ω can be carried out in linear time, as they correspond to simple shifts and negations.

The Schönhage–Strassen algorithm remained the champion for more than thirty years, before being superseded by Fürer's algorithm [12]. In short, Fürer managed to combine the advantages of the two algorithms from [49], to achieve the bound (1.1) for some

unspecified constant $K_{\mathbb{Z}}$. In [25, section 7], it has been shown that an optimized version of Fürer's original algorithm achieves $K_{\mathbb{Z}} = 16$. In a succession of works [25, 18, 19, 21], new algorithms were proposed for which $K_{\mathbb{Z}} = 8$, $K_{\mathbb{Z}} = 6$, $K_{\mathbb{Z}} = 4\sqrt{2}$, and $K_{\mathbb{Z}} = 4$. In view of the companion paper [22], we now know that $\mathsf{I}(n) = O(n \log n)$.

Historically speaking, we also notice that improvements on the lowest possible values of $K_{\mathbb{Z}}$ and $K_{\mathbb{F}}$ were often preceded by improvements modulo suitable number theoretic assumptions. In [25, section 9], we proved that one may take $K_{\mathbb{Z}} = 4$ under the assumption that there exist "sufficiently many" Mersenne primes [25, Conjecture 9.1]. The same value $K_{\mathbb{Z}} = 4$ was achieved in [10] modulo the existence of sufficiently many generalized Fermat primes. In [20], we again reached $K_{\mathbb{Z}} = 4$ under the assumption that $P(k) = O(\varphi(k) \log^2 k) = O(k \log^2 k)$, where φ is the Euler totient function.

Polynomial multiplication. To a large extent, the development of more efficient algorithms for polynomial multiplication went hand in hand with progress on integer multiplication. The early algorithms by Karatsuba and Toom [34, 56], as well as Schönhage–Strassen's second algorithm from [49], are all based on increasingly efficient algebraic methods for polynomial multiplication over more or less general rings R.

More precisely, let $\mathsf{M}_R^{\mathrm{alg}}(n)$ be the number of ring operations in R that are required for the multiplication of two polynomials of degree $<n$ in $R[x]$. A straightforward adaptation of Karatsuba's algorithm for integer multiplication gives rise to the algebraic complexity bound $\mathsf{M}_R^{\mathrm{alg}}(n) = O(n^{\log 3/\log 2})$.

The subsequent methods typically require mild hypotheses on R: since Toom's algorithm uses more than three evaluation points, its polynomial analogue requires R to contain sufficiently many points. Similarly, Schönhage–Strassen multiplication is based on "dyadic" FFTs and therefore requires 2 to be invertible in R. Schönhage subsequently developed a "triadic" analogue of their method that is suitable for polynomial multiplication over fields of characteristic two [48]. The bound $\mathsf{M}_R^{\mathrm{alg}}(n) = O(n \log n \log\log n)$ for general (not necessarily commutative) rings R is due to Cantor and Kaltofen [8].

Let us now turn to the bit complexity $\mathsf{M}_q(n)$ of polynomial multiplication over a finite field $\mathbb{F}_q$ with $q = \pi^\kappa$ and where π is prime. Using Kronecker substitution, it is not hard to show that

$$\mathsf{M}_{\pi^\kappa}(n) \leqslant \mathsf{M}_\pi(2n\kappa) + O(n\,\mathsf{M}_\pi(\kappa)),$$

which allows us to reduce our study to the particular case when $q = \pi$ and $\kappa = 1$.

Now the multiplication of polynomials in $\mathbb{F}_\pi[x]$ of small degree can be reduced to integer multiplication using Kronecker substitution: the input polynomials are first lifted into polynomials with integers coefficients in $\{0, \ldots, \pi - 1\}$, and then evaluated at $x = 2^{\lceil \log_2(n\pi^2) \rceil}$. The desired result can finally be read off from the integer product of these evaluations. If $\log n = O(\log \pi)$, this yields

$$\mathsf{M}_\pi(n) = O(\mathsf{I}(n \log \pi)). \tag{1.4}$$

On the other hand, for $\log \pi = o(\log n)$, adaptation of the algebraic complexity bound $\mathsf{M}_R^{\mathrm{alg}}(n) = O(n \log n \log\log n)$ to the Turing model yields

$$\mathsf{M}_\pi(n) = O(n \log n \log\log n \log \pi + n \log n\, \mathsf{I}(\log \pi)), \tag{1.5}$$

where the first term corresponds to additions/subtractions in $\mathbb{F}_\pi$ and the second one to multiplications. Notice that the first term dominates for large n. The combination of (1.4) and (1.5) also implies

$$\mathsf{M}_\pi(n) \;=\; O(n \log \pi \, (\log(n \log \pi))^{1+o(1)}). \tag{1.6}$$

For almost forty years, the bound (1.5) remained unbeaten. Since Fürer's algorithm (alike Schönhage–Strassen's first algorithm from [49]) essentially relies on the availability of suitable roots of unity in the base field R, it admits no direct algebraic analogue. In particular, the existence of a Fürer-type bound of the form (1.2) remained open for several years. This problem got first solved in [27] for $K_{\mathbb{F}} = 8$, but using an algorithm that is very different from Fürer's algorithm for integer multiplication. Under suitable number theoretic assumptions, it was also shown in the preprint version [24] that one may take $K_{\mathbb{F}} = 4$, a result that was achieved subsequently without these assumptions [23].

Let us finally notice that it usually a matter of routine to derive better polynomial multiplication algorithms over $\mathbb{Z}/m\mathbb{Z}$ for integers $m > 0$ from better multiplication algorithms over $\mathbb{F}_\pi$. These techniques are detailed in [27, sections 8.1–8.3]. Denoting by $\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}(n)$ the bit complexity of multiplying two polynomials of degree $<n$ over $\mathbb{Z}/m\mathbb{Z}$, it is shown there that

$$\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}(n) \;=\; O\big(n \log m \log (n \log m) \, K_{\mathbb{F}}^{\log^*(n \log m)}\big), \tag{1.7}$$

uniformly in m, and for $K_{\mathbb{F}} = 8$. Exploiting the fact that finite fields only contain a finite number of elements, it is also a matter of routine to derive algebraic complexity bounds for $\mathsf{M}_R^{\mathrm{alg}}(n)$ in the case when R is a ring of finite characteristic. We refer to [27, sections 8.4 and 8.5] for more details.

Related tools. In order to prove Theorems 1.1 and 1.2, we will make use of several other tools from the theory of discrete Fourier transforms (DFTs). First of all, given a ring R and a composite integer $n = n_1 \cdots n_d$ such that the n_i are mutually coprime, the Chinese remainder theorem gives rise to an isomorphism

$$R[x]/(x^n - 1) \;\cong\; R[x_1, \ldots, x_d]/(x_1^{n_1} - 1, \ldots, x_d^{n_d} - 1).$$

This was first observed in [1] and we will call the corresponding conversions *CRT transforms*. The above isomorphism also allows for the reduction of a univariate DFT of length n to a multivariate DFT of length $n_1 \times \cdots \times n_d$. This observation is older and due to Good [15] and independently to Thomas [55].

Another classical tool from the theory of discrete Fourier transforms is Rader reduction [46]: a DFT of prime length p over a ring R can be reduced to the computation of a cyclic convolution of length $p-1$, i.e. a product in the ring $R[x]/(x^{p-1} - 1)$. In section 4, we will also present a multivariate version of this reduction. In a similar way, one may use Bluestein's chirp transform [4] to reduce a DFT of general length n over R to a cyclic convolution of the same length. Whereas FFT-multiplication is a technique that reduces polynomial multiplications to the computation of DFTs, Bluestein's algorithm (as well as Rader's algorithm for prime lengths) can be used for reductions in the opposite direction. In particular, Theorem 1.2 implies that a DFT of length n over a finite field $\mathbb{F}_q$ can be computed in time $O(\mathsf{M}_q(n)) = O(n \log q \log(n \log q))$ on a Turing machine.

Nussbaumer polynomial transforms [42, 43] constitute yet another essential tool for our new results. In a similar way as in Schönhage–Strassen's second algorithm from [49], the idea is to use DFTs over a ring $R = C[u]/(u^N + 1)$ with $N \in 2^{\mathbb{N}}$. This ring has the property that u is a $2N$-th principal root of unity and that multiplications by powers of u can be computed in linear time. In particular, DFTs of length $n \mid 2N$ can be computed very efficiently using only additions and subtractions. Nussbaumer's important observation is that such transforms are especially interesting for the computation of multidimensional DFTs (in [49] they are only used for univariate DFTs). Now the lengths of these multidimensional DFTs should divide $2N$ in each direction. Following a suggestion by Nussbaumer and Quandalle in [44, p. 141], this situation can be forced using Rader reduction and CRT transforms.

1.2. Overview of the new results

In order to prove Theorems 1.1 and 1.2, we make use of a combination of well-known techniques from the theory of fast Fourier transforms. We just surveyed these related tools; more details are provided in sections 2, 3 and 4. Notice that part of section 2 contains material that we adapted from previous papers [25, 27] and included for convenience of the reader. In section 5, the assumption on the existence of small Linnik constants L will also be discussed in more detail, together with a few consequences. Throughout this paper, all computations are done in the deterministic multitape Turing model [16] and execution times are analyzed in terms of bit complexity. The appendix covers technical details about the cost of data rearrangements when using this model.

Integer multiplication. We prove Theorem 1.1 in section 6. The main idea of our new algorithm is to reduce integer multiplication to the computation of multivariate cyclic convolutions in a suitable algebra of the form

$$\begin{aligned} \mathcal{R} &= \mathbb{A}[x_1, \ldots, x_d]/(x_1^{p_1} - 1, \ldots, x_d^{p_d} - 1) \\ \mathbb{A} &= (\mathbb{Z}/m\mathbb{Z})[u]/(u^s + 1). \end{aligned}$$

Here s is a power of two and $p_1, \ldots, p_d$ are the first d prime numbers in the arithmetic progression $\ell + 1, 3\ell + 1, 5\ell + 1, \ldots$, where ℓ is another power of two with $s^{1/3} \leqslant \ell \leqslant s^{2/3}$. Setting $\nu = \operatorname{lcm}(\ell^3, p_1 - 1, \ldots, p_d - 1)\, p_1 \cdots p_d$, we choose m such that there exists a principal ν-th root of unity in $\mathbb{Z}/m\mathbb{Z}$ that makes it possible to compute products in the algebra $\mathcal{R}$ using fast Fourier multiplication. Concerning the dimension d, it turns out that we may take $d = O(1)$, although larger dimensions may allow for speed-ups by a constant factor as long as $\operatorname{lcm}(\ell^3, p_1 - 1, \ldots, p_d - 1)$ can be kept small.

Using multivariate Rader transforms, the required discrete Fourier transforms in $\mathcal{R}$ essentially reduce to the computation of multivariate cyclic convolutions in the algebra

$$\mathcal{S} = \mathbb{A}[z_1, \ldots, z_d]/\left(z_1^{p_1 - 1} - 1, \ldots, z_d^{p_d - 1} - 1\right).$$

By construction, we may factor $p_i - 1 = \ell q_i$, where q_i is an odd number that is small due the hypothesis on the Linnik constant. Since $q_i \mid \nu$, we notice that $\mathbb{Z}/m\mathbb{Z}$ contains a primitive q_i-th root of unity. CRT transforms allow us to rewrite the algebra $\mathcal{S}$ as

$$\begin{aligned} \mathcal{S} &\cong \mathbb{B}[y_1, \ldots, y_d]/(y_1^{\ell} - 1, \ldots, y_d^{\ell} - 1) \\ \mathbb{B} &= \mathbb{A}[v_1, \ldots, v_d]/(v_1^{q_1} - 1, \ldots, v_d^{q_d} - 1). \end{aligned}$$

Now the important observation is that u can be considered as a "fast" principal $(2s)$-th root of unity in both $\mathbb{A}$ and $\mathbb{B}$. This means that products in $\mathcal{S}$ can be computed using multivariate Fourier multiplication with the special property that the discrete Fourier transforms become Nussbaumer polynomial transforms. Since s is a power of two, these transforms can be computed in time $O(n \log n)$. Moreover, for sufficiently small Linnik constants L, the cost of the "inner multiplications" in $\mathbb{B}$ can be mastered so that it only marginally contributes to the overall cost.

Polynomial multiplication. In order to prove Theorem 1.2, we proceed in a similar manner; see sections 7 and 8. It suffices to consider the case that $q = \pi$ is prime. In section 7, we first focus on ground fields of characteristic $\pi > 2$. This time, the ring $\mathbb{Z}/m\mathbb{Z}$ needs to be replaced by a suitable extension field $\mathbb{F}_{\pi^\kappa}$ of $\mathbb{F}_\pi$; in particular, we define $\mathbb{A} = \mathbb{F}_{\pi^\kappa}[u]/(u^s + 1)$. More precisely, we take $\kappa = \operatorname{lcm}(p_1 - 1, \ldots, p_d - 1)$, which ensures the existence of primitive $(p_1 \cdots p_d)$-th roots of unity in $\mathbb{F}_{\pi^\kappa}$ and $\mathbb{A}$.

The finite field case gives rise to a few additional technical difficulties. First of all, the bit size of an element of $\mathbb{F}_{\pi^\kappa}$ is exponentially larger than the bit size of an element of $\mathbb{Z}/m\mathbb{Z}$. This makes the FFT-approach for multiplying elements in $\mathbb{B}$ not efficient enough. Instead, we impose the additional requirement that $q_1, \ldots, q_d$ be pairwise coprime. This allows us to reduce multiplications in $\mathbb{B}$ to univariate multiplications in $\mathbb{A}[v]/(v^{q_1 \cdots q_d} - 1)$, but at the expense of the much stronger hypothesis $L < 1 + 2^{-1162}$ on L.

A second technical difficulty concerns the computation of a defining polynomial for $\mathbb{F}_{\pi^\kappa}$ (i.e. an irreducible polynomial μ in $\mathbb{F}_\pi[x]$ of degree κ), together with a primitive $(p_1 \cdots p_d)$-th root of unity $\omega \in \mathbb{F}_{\pi^\kappa}$. For our choice of κ as a function of n, and thanks to work by Shoup [51, 52], it turns out that both μ and ω can be precomputed in linear time $O(n \lg \pi)$.

The case when $\pi = 2$ finally comes with its own particular difficulties, so we treat it separately in section 8. Since 2 is no longer invertible, we cannot use DFT lengths ℓ that are powers of two. Following Schönhage [48], we instead resort to "triadic" FFTs, for which ℓ becomes a power of three. More annoyingly, since $p_i - 1$ is necessarily even for $i = 1, \ldots, d$, this also means that we now have to take $p_i - 1 = 2\ell q_i$. In a addition to the multivariate cyclic convolutions of lengths $(\ell, \overset{d\times}{\ldots}, \ell)$ and $(q_1, \ldots, q_d)$ from before, this gives rise to multivariate cyclic convolutions of length $(2, \overset{d\times}{\ldots}, 2)$, which need to be handled separately. Although the proof of Theorem 1.2 in the case when $\pi = 2$ is very similar to the case when $\pi \neq 2$, the above complications lead to subtle changes in the choices of parameters and coefficient rings. For convenience of the reader, we therefore reproduced most of the proof from section 7 in section 8, with the required adaptations.

Variants. The constants $1 + 1/303$ and 2^{-1162} in Theorems 1.1 and 1.2 are not optimal: we have rather attempted to keep the proofs as simple as possible. Nevertheless, our proofs do admit two properties that deserve to be highlighted:

- The hypothesis that there exists a Linnik constant that is sufficiently close to one is enough for obtaining our main complexity bounds.
- The dimension d of the multivariate Fourier transforms can be kept bounded and does not need to grow with n.

We refer to Remarks 6.1 and 7.1 for some ideas on how to optimize the thresholds for acceptable Linnik constants.

Our main algorithms admit several variants. It should for instance be relatively straightforward to use approximate complex arithmetic in section 6 instead of modular arithmetic. Due to the fact that primitive roots of unity in $\mathbb{C}$ are particularly simple, the Linnik constants may also become somewhat better, but additional care is needed for the numerical error analysis. Using similar ideas as in [27, section 8], our multiplication algorithm for polynomials over finite fields can also be adapted to polynomials over finite rings and more general effective rings of positive characteristic.

Another interesting question is whether there exist practically efficient variants that might outperform existing implementations. Obviously, this would require further fine-tuning, since the "crazy" constants in our proofs were not motivated by practical applications. Nevertheless, our approach combines several ideas from existing algorithms that have proven to be efficient in practice. In the case of integer multiplication, this makes us more optimistic than for the previous post-Fürer algorithms [12, 13, 11, 25, 10, 18, 19, 21]. In the finite field case, the situation is even better, since similar ideas have already been validated in practice [26, 32].

Applications. Assume that there exist Linnik constants that are sufficiently close to 1. Then Theorems 1.1 and 1.2 have many immediate consequences, as many computational problems may be reduced to integer or polynomial multiplication.

For example, Theorem 1.1 implies that the Euclidean division of two n-bit integers can be performed in time $O(\mathsf{I}(n)) = O(n \log n)$, whereas the gcd can be computed in time $O(\mathsf{I}(n) \log n) = O(n \log^2 n)$. Many transcendental functions and constants can also be approximated faster. For instance, n binary digits of π can be computed in time $O(\mathsf{I}(n) \log n) = O(n \log^2 n)$ and the result can be converted into decimal notation in time $O(\mathsf{I}(n) \log n / \log\log n) = O(n \log^2 n / \log\log n)$. For details and more examples, we refer to [7, 31].

In a similar way, Theorem 1.2 implies that the Euclidean division of two polynomials in $\mathbb{F}_q$ with $q = \pi^\kappa$ of degree n can be performed in time $O(\mathsf{M}_q(n)) = O(n \log q \log(n \log q))$, whereas the extended gcd admits bit-complexity $O(\mathsf{M}_q(n) \log n) = O(n \log n \log q \log(n \log q))$. Since inversion of a non-zero element of $\mathbb{F}_q$ boils down to an extended gcd computation of degree κ over $\mathbb{F}_q$, this operation admits bit complexity $O(\kappa \log \kappa \log \pi \log (\kappa \log \pi))$. The complexities of many operations on formal power series over $\mathbb{F}_q$ can also be expressed in terms of $\mathsf{M}_q(n)$. We refer to [14, 30] for details and more applications.

2. UNIVARIATE ARITHMETIC

2.1. Basic notations

Let R be a commutative ring with identity. Given $n \in \mathbb{N}^>$, we define

$$\begin{aligned} R[x]_n &:= \{P \in R[x] : \deg P < n\} \\ R[x]_n^\circ &:= R[x]/(x^n - 1) \\ R[x]_n^\ominus &:= R[x]/(x^n + 1). \end{aligned}$$

Elements of $R[x]_n^\circ$ and $R[x]_n^\ominus$ will be called *cyclic polynomials* and *negacyclic polynomials*, respectively. For subsets $S \subseteq R$, it will be convenient to write $S[x] := \{a_d x^d + \cdots + a_0 \in R[x] : a_0, \ldots, a_d \in S\}$, and extend the above notations likewise. This is typically useful in the case when $R = A[u]$ for some other ring A and $S = A[u]_k$ for some $k \in \mathbb{N}^>$.

In our algorithms, we will only consider *effective rings* R, whose elements can be represented using data structures of a fixed bitsize s_R and such that algorithms are available for the ring operations. We will always assume that additions and subtractions can be computed in linear time $O(\mathsf{s}_R)$ and we denote by m_R the bit complexity of multiplication in R. For some fixed invertible integer n, we sometimes need to divide elements of R by n; we define $\mathsf{d}_{R,\mathbb{Z}}$ to be the cost of such a division (assuming that a pre-inverse of n has been precomputed). If $R = \mathbb{Z}/m\mathbb{Z}$ with $m = \pi^\lambda$ and π prime, then we have $\mathsf{s}_R = \lg m \leqslant \lambda \lg \pi$, $\mathsf{m}_R = O(\mathsf{I}(\lg m))$, and $\mathsf{d}_{R,\mathbb{Z}} = O(\mathsf{I}(\lg m))$, where $\lg x := \lceil \log x / \log 2 \rceil$.

We write $\mathsf{M}_R(n)$ for the bit complexity of multiplying two polynomials in $R[x]_n$. We also define $\mathsf{M}_R^\circ(n) := \mathsf{m}_{R[x]_n^\circ}$ and $\mathsf{M}_R^\ominus(n) := \mathsf{m}_{R[x]_n^\ominus}$. Multiplications in $R[x]_n^\circ$ and $R[x]_n^\ominus$ are also called *cyclic convolutions* and *negacyclic convolutions*. Since reduction modulo $x^n \pm 1$ can be performed using n additions, we clearly have

$$\mathsf{M}_R^\circ(n) \leqslant \mathsf{M}_R(n) + O(n\,\mathsf{s}_R) \tag{2.1}$$
$$\mathsf{M}_R^\ominus(n) \leqslant \mathsf{M}_R(n) + O(n\,\mathsf{s}_R). \tag{2.2}$$

We also have

$$\mathsf{M}_R(n) \leqslant \mathsf{M}_R^\circ(2n) \tag{2.3}$$
$$\mathsf{M}_R(n) \leqslant \mathsf{M}_R^\ominus(2n), \tag{2.4}$$

since $PQ \in R[x]_{2n}$ for all $P, Q \in R[x]_n$.

In this paper we will frequently convert between various representations. Given a computable ring morphism φ between two effective rings R and S, we will write $\mathsf{C}(\varphi)$ for the cost of applying φ to an element of R. If φ is an embedding of R into S that is clear from the context, then we will also write $\mathsf{C}(R \to S)$ instead of $\mathsf{C}(\varphi)$. If R and S are isomorphic, with morphisms φ and φ^{-1} for performing the conversions that are clear from context, then we define $\mathsf{C}(R \leftrightarrow S) := \max(\mathsf{C}(R \to S), \mathsf{C}(S \to R))$.

2.2. Discrete Fourier transforms and fast Fourier multiplication

Let R be a ring with identity and let $\omega \in R$ be such that $\omega^n = 1$ for some $n \in \mathbb{N}^>$. Given a vector $a = (a_0, \ldots, a_{n-1}) \in R^n$, we may form the polynomial $A = a_0 + \cdots + a_{n-1} x^{n-1} \in R[x]_n$, and we define the *discrete Fourier transform* of a with respect to ω to be the vector $\hat{a} = \mathrm{DFT}_\omega(a) \in R^n$ with $\hat{a}_i = A(\omega^i)$ for all $i \in \mathbb{N}_n := \{0, \ldots, n-1\}$.

Assume now that ω is a *principal n-th root of unity*, meaning that

$$\sum_{k=0}^{n-1} (\omega^i)^k = 0 \tag{2.5}$$

for all $i \in \{1, \ldots, n-1\}$. Then $\omega^{-1} = \omega^{n-1}$ is again a principal n-th root of unity, and we have

$$\mathrm{DFT}_{\omega^{-1}}(\mathrm{DFT}_\omega(a)) = n\,a. \tag{2.6}$$

Indeed, writing $b = \mathrm{DFT}_{\omega^{-1}}(\mathrm{DFT}_\omega(a))$, the relation (2.5) implies that

$$b_i = \sum_{j=0}^{n-1} \hat{a}_j \omega^{-ji} = \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} a_k \omega^{j(k-i)} = \sum_{k=0}^{n-1} a_k \sum_{j=0}^{n-1} \omega^{j(k-i)} = \sum_{k=0}^{n-1} a_k (n\,\delta_{i,k}) = n\,a_i,$$

where $\delta_{i,k} = 1$ if $i = k$ and $\delta_{i,k} = 0$ otherwise.

Consider any n-th root of unity ω^i. Given a cyclic polynomial $A \in R[x]_n^\circ$ that is presented as the class of a polynomial $\tilde{A} \in R[x]$ modulo $x^n - 1$, the value $\tilde{A}(\omega^i)$ does not depend on the choice of $\tilde{A}$. It is therefore natural to consider $A(\omega^i) := \tilde{A}(\omega^i)$ as the evaluation of A at ω^i. It is now convenient to extend the definition of the discrete Fourier transform and set $\mathrm{DFT}_\omega(A) := (A(1), A(\omega), \ldots, A(\omega^{n-1})) \in R^n$. If n is invertible in R, then DFT_ω becomes a ring isomorphism between the rings $R[x]_n^\circ$ and R^n. Indeed, for any A, $B \in R[x]_n^\circ$ and $i \in \{0, \ldots, n-1\}$, we have $(A\,B)(\omega^i) = A(\omega^i)\,B(\omega^i)$. Furthermore, the relation (2.6) provides us with a way to compute the inverse transform DFT_ω^{-1}.

In particular, denoting by $\mathsf{F}_R(n, \omega)$ the cost of computing a discrete Fourier transform of length n with respect to ω, we get

$$\begin{aligned} \mathsf{C}(R[x]_n^\circ \to R^n) &\leqslant \mathsf{F}_R(n, \omega) \\ \mathsf{C}(R^n \to R[x]_n^\circ) &\leqslant \mathsf{F}_R(n, \omega^{-1}) + n\,\mathsf{d}_{R,\mathbb{Z}}. \end{aligned}$$

Since DFT_ω and $\mathrm{DFT}_{\omega^{-1}}$ coincide up to a simple permutation, we also notice that $\mathsf{F}_R(n, \omega^{-1}) \leqslant \mathsf{F}_R(n, \omega) + O(n\,\mathsf{s}_R)$. Setting $\mathsf{F}_R(n) := \mathsf{F}_R(n, \omega)$ for some "privileged" principal n-th root of unity $\omega \in R$ (that will always be clear from the context), it follows that $\mathsf{C}(R[x]_n^\circ \leftrightarrow R^n) \leqslant \mathsf{F}_R(n) + n\,\mathsf{d}_{R,\mathbb{Z}} + O(n\,\mathsf{s}_R)$.

A well known application of this isomorphism is to compute cyclic convolutions $A\,B$ for $A, B \in R[x]_n^\circ$ in R^n using the formula

$$A\,B \;=\; \mathrm{DFT}_\omega^{-1}(\mathrm{DFT}_\omega(A)\,\mathrm{DFT}_\omega(B)).$$

It follows that

$$\begin{aligned} \mathsf{M}_R^\circ(n) &\leqslant 2\,\mathsf{C}(R[x]_n^\circ \to R^n) + \mathsf{m}_{R^n} + \mathsf{C}(R^n \to R[x]_n^\circ) \\ &\leqslant 3\,\mathsf{F}_R(n) + n\,(\mathsf{m}_R + \mathsf{d}_{R,\mathbb{Z}}) + O(n\,\mathsf{s}_R). \end{aligned}$$

Computing cyclic convolutions in this way is called *fast Fourier multiplication*.

2.3. The Cooley–Tukey FFT

Let ω be a principal n-th root of unity and let $n = n_1\,n_2$ where $1 < n_1 < n$. Then ω^{n_1} is a principal n_2-th root of unity and ω^{n_2} is a principal n_1-th root of unity. Moreover, for any $i_1 \in \{0, \ldots, n_1 - 1\}$ and $i_2 \in \{0, \ldots, n_2 - 1\}$, we have

$$\begin{aligned} \hat{a}_{i_1 n_2 + i_2} &= \sum_{k_1=0}^{n_1-1} \sum_{k_2=0}^{n_2-1} a_{k_2 n_1 + k_1} \omega^{(k_2 n_1 + k_1)(i_1 n_2 + i_2)} \\ &= \sum_{k_1=0}^{n_1-1} \omega^{k_1 i_2} \left(\sum_{k_2=0}^{n_2-1} a_{k_2 n_1 + k_1} (\omega^{n_1})^{k_2 i_2} \right) (\omega^{n_2})^{k_1 i_1}. \end{aligned} \tag{2.7}$$

If $\mathcal{A}_1$ and $\mathcal{A}_2$ are algorithms for computing DFTs of length n_1 and n_2, then we may use (2.7) to construct an algorithm for computing DFTs of length n as follows.

For each $k_1 \in \{0,\ldots,n_1-1\}$, the sum inside the brackets corresponds to the i_2-th coefficient of a DFT of the n_2-tuple $(a_{0n_1+k_1},\ldots,a_{(n_2-1)n_1+k_1}) \in R^{n_2}$ with respect to ω^{n_1}. Evaluating these *inner DFTs* requires n_1 calls to $\mathcal{A}_2$. Next, we multiply by the *twiddle factors* $\omega^{k_1 i_2}$, at a cost of n operations in R. Finally, for each $i_2 \in \{0,\ldots,n_2-1\}$, the outer sum corresponds to the i_1-th coefficient of a DFT of an n_1-tuple in R^{n_1} with respect to ω^{n_2}. These *outer DFTs* require n_2 calls to $\mathcal{A}_1$. On a Turing machine, the application of the n_1 inner DFTs also requires us to reorganize a into n_1 consecutive vectors of size n_2, and *vice versa*. This can be done in time $O(n \lg \min(n_1,n_2)\, \mathsf{s}_R)$ using matrix transpositions (see section A.1 in the appendix).

Let $\mathsf{F}^*_R(n)$ denote the cost of performing a DFT of length n, while assuming that the twiddle factors $\omega^0, \omega^1, \ldots, \omega^{n-1}$ have been precomputed. Then computing the DFT using the above Cooley–Tukey algorithm yields the bound

$$\mathsf{F}^*_R(n_1 n_2) \;\leqslant\; n_1 \mathsf{F}^*_R(n_2) + n_2 \mathsf{F}^*_R(n_1) + n\,\mathsf{m}_R + O(n \lg \min(n_1,n_2)\, \mathsf{s}_R).$$

For a factorization $n = n_1 \cdots n_d$, an easy induction over d then yields

$$\mathsf{F}^*_R(n) \;\leqslant\; \sum_{i=1}^{d} \frac{n}{n_i} \mathsf{F}^*_R(n_i) + (d-1)\, n\, \mathsf{m}_R + O(n \lg n\, \mathsf{s}_R).$$

The twiddle factors can obviously be computed in time $n\,\mathsf{m}_R + O(n\,\mathsf{s}_R)$. Altogether, we obtain

$$\mathsf{F}_R(n) \;\leqslant\; \sum_{i=1}^{d} \frac{n}{n_i} \mathsf{F}_R(n_i) + d\, n\, \mathsf{m}_R + O(n \lg n\, \mathsf{s}_R). \tag{2.8}$$

2.4. Nussbaumer polynomial transforms

In the case that $R = C[u]^{\ominus}_N$ for some effective ring C and power of two $N \in 2^{\mathbb{N}}$, the indeterminate u is itself a principal $(2N)$-th root of unity. Moreover, multiplications by powers of u can be computed in linear time $O(\mathsf{s}_R) = O(N\,\mathsf{s}_C)$ since

$$(a_0 + \cdots + a_{N-1} u^{N-1})\, u^k \;=\; -a_{N-k} - \cdots - a_{N-1} u^{k-1} + a_0 u^k + \cdots + a_{N-k-1} u^{N-1}$$

for all $a_0 + \cdots + a_{N-1} u^{N-1} \in R$ and $0 \leqslant k < N$. For this reason, u is sometimes called a *fast principal root of unity*.

Now assume that we wish to compute a discrete Fourier transform of size $n \mid 2N$ over R. Then we may use the Cooley–Tukey algorithm from the previous subsection with $\omega = u^{2N/n}$, $d = \lg n$, and $n_1 = \cdots = n_d = 2$. When using our faster algorithm for multiplications by twiddle factors, the term $d\,n\,\mathsf{m}_R$ in (2.8) can be replaced by $O(d\,n\,N\,\mathsf{s}_C) = O(d\,n\,\mathsf{s}_R) = O(n \lg n\, \mathsf{s}_R)$. Moreover, discrete Fourier transforms of length 2 just map pairs $(a,b) \in R^2$ to pairs $(a+b, a-b) \in R^2$, so they can also be computed in linear time $O(\mathsf{s}_R)$. Altogether, the bound (2.8) thus simplifies into

$$\mathsf{F}_R(n) \;=\; O(n \lg n\, \mathsf{s}_R).$$

In terms of scalar operations in C, this yields

$$\mathsf{F}_{C[u]^{\ominus}_N}(n) \;=\; O(N n \lg n\, \mathsf{s}_C).$$

Discrete Fourier transforms over rings R of the type considered in this subsection are sometimes qualified as *Nussbaumer polynomial transforms*.

2.5. Triadic Nussbaumer polynomial transforms

If 2 is invertible in the effective ring C from the previous subsection, then so is n, and we can compute the inverse DFT of length n using

$$\mathrm{DFT}^{-1}_{u^{2N/n}}(a) \;=\; \frac{1}{n}\mathrm{DFT}_{u^{-2N/n}}(a).$$

However, if C has characteristic 2 (i.e. $1+1=0$ in C), then this is no longer possible, and we need to adapt the method if we want to use it for polynomial multiplication.

Following Schönhage [48], the remedy is to use triadic DFTs. More precisely, assuming that $N \in 2\cdot 3^{\mathbb{N}}$, we now work over the ring $R = C[u]^{\circ}_N$ instead of $R = C[u]^{\ominus}_N$, where

$$C[u]^{\circ}_N \;:=\; C[u]/(u^N + u^{N/2} + 1).$$

It is easily verified that u is a principal $(3N/2)$-th root of unity in $C[u]^{\circ}_N$. Moreover, multiplications of elements in $C[u]^{\circ}_N$ with powers of u can still be computed in linear time $O(N\,\mathsf{s}_C)$. Indeed, given $a = a_0 + \cdots + a_{N-1}u^{N-1}$ and $k \in \{0,\ldots,N/2\}$, we have

$$a\,u^k \;=\; (a_{N-k} + \cdots + a_{N-1}u^{k-1} + a_0 u^k + \cdots + a_{N-1-k}u^{N-1}) + (a_{N-k}u^{N/2} + \cdots + a_{N-1}u^{N/2+k-1}).$$

Multiplications with $u^{k+N/2}$ (resp. u^{k+N}) can be reduced to this case, by regarding them as two multiplications with u^k and $u^{N/2}$ (resp. three multiplications with u^k, $u^{N/2}$, and $u^{N/2}$). Using similar formulas, polynomials in $C[u]$ of degree $<d$ can be reduced in linear time $O(d\,\mathsf{s}_C)$ modulo $u^N + u^{N/2} + 1$.

Now assume that we wish to compute a discrete Fourier transform of size $n \mid (3N/2)$ over R. Discrete Fourier transforms of length 3 now map triples $(a,b,c) \in R^3$ to triples $(a+b+c,\, a+b\,u^{N/2}+c\,u^N,\, a+b\,u^N+c\,u^{N/2}) \in R^3$, so they can be computed in linear time $O(\mathsf{s}_R)$. Using the Cooley–Tukey algorithm with $\omega = u^{3N/(2n)}$, $d = \log_3 n$, and $n_1 = \cdots = n_d = 3$, similar considerations as in the previous subsection again allows us to conclude that

$$\begin{aligned}\mathsf{F}_R(n) \;&=\; O(n \lg n\, \mathsf{s}_R)\\ \mathsf{F}_{C[u]^{\circ}_N}(n) \;&=\; O(N n \lg n\, \mathsf{s}_C).\end{aligned}$$

We will qualify discrete Fourier transforms over rings R of this type as *triadic Nussbaumer polynomial transforms*.

2.6. Bluestein's chirp transform

We have shown above how to multiply polynomials using DFTs. Inversely, it is possible to reduce the computation of DFTs — of arbitrary length, not necessarily a power of two — to polynomial multiplication [4], as follows.

Let ω be a principal n-th root of unity. If n is even, then we assume that there exists some $\eta \in R$ with $\eta^2 = \omega$. If n is odd, then we take $\eta = \omega^{(n+1)/2}$, so that $\eta^2 = \omega$ and $\eta^n = 1$. Consider the sequences

$$f_i = \eta^{i^2}, \quad g_i = \eta^{-i^2}.$$

Then $\omega^{ij} = f_i f_j g_{i-j}$, so for any $a \in R^n$ we have

$$\hat{a}_i = \sum_{j=0}^{n-1} a_j \omega^{ij} = f_i \sum_{j=0}^{n-1} (a_j f_j)\, g_{i-j}. \tag{2.9}$$

If n is even, then we have

$$g_{i+n} = \eta^{-(i+n)^2} = \eta^{-i^2-n^2-2ni} = \eta^{-i^2}\,\omega^{-(\frac{n}{2}+i)n} = g_i.$$

If n is odd, then we also have

$$g_{i+n} = \eta^{-(i+n)^2} = \eta^{-i^2-n^2-2ni} = \eta^{-i^2}\,(\eta^n)^{-n-2i} = g_i.$$

Now let $F = f_0 a_0 + \cdots + f_{n-1} a_{n-1} X^{n-1}$, $G = g_0 + \cdots + g_{n-1} X^{n-1}$ and $C = c_0 + \cdots + c_{n-1} X^{n-1} \equiv FG$ modulo $X^n - 1$. Then (2.9) implies that $\hat{a}_i = f_i c_i$ for all $i \in \{0, \ldots, n-1\}$. In other words, the computation of a DFT of even length n reduces to a cyclic convolution product of the same length, together with $O(n)$ additional operations in R:

$$\mathsf{F}_R(n) \;\leqslant\; \mathsf{M}_R^{\circ}(n) + O(n\,\mathsf{m}_R). \tag{2.10}$$

Notice that the polynomial G is fixed and independent of a in this product. The f_i (and g_i) can be computed in time $O(n\,\mathsf{m}_R)$ *via* the identity $f_{i+1} = \eta^{2i+1} f_i$.

3. MULTIVARIATE ARITHMETIC

3.1. Tensor products

It is well known that multivariate Fourier transforms can be interpreted as tensor products of univariate Fourier transforms. Since we will also need another type of multivariate "Rader transform" in the next section, it will be useful to recall this point of view. We refer to [37, Chapter XVI] for an abstract introduction to tensor products.

Let R be a commutative ring and let A and B be two R-modules. The *tensor product* $A \otimes B$ of A and B is an R-module together with a bilinear mapping $\otimes : A \times B \to A \otimes B$. This bilinear mapping satisfies the universal property that for any other R-module C together with a bilinear mapping $\varphi : A \times B \to C$, there exists a unique linear mapping $\xi : A \otimes B \to C$ with $\varphi = \xi \circ \otimes$. Up to natural isomorphisms, it can be shown that the tensor product operation $\otimes$ is associative and distributive with respect to the direct sum operation $\oplus$.

Besides modules, it is also possible to tensor linear mappings. More precisely, given two linear mappings $\varphi : A \to M$ and $\psi : B \to N$, the mapping that sends $(a, b) \in A \times B$ to $\varphi(a) \otimes \psi(b)$ is clearly bilinear. This means that there exists a unique mapping $\xi : A \otimes B \to M \otimes N$ such that $\varphi(a) \otimes \psi(b) = \xi(a \otimes b)$ for all $(a, b) \in A \times B$. We call $\varphi \otimes \psi := \xi$ the *tensor product* of φ and ψ.

Assume now that A and B are free R-modules of finite rank, say $A = R^a$ and $B = R^b$. Then $A \otimes B$ is again a free R-module that can be identified with the set $R^{a \times b}$ of bidimensional arrays of size $a \times b$ and with coefficients in R. The tensor product of two vectors $x = (x_1, \ldots, x_a) \in A$ and $y = (y_1, \ldots, y_b) \in B$ is given by the array $x \otimes y$ with entries $(x \otimes y)_{i,j} = x_i y_j$.

Let $M = R^m$ and $N = R^n$ be two more free R-modules of finite rank. Assume that we are given two linear mappings $\varphi : A \to M$ and $\psi : B \to N$ that can be computed by an algorithm. Then $\varphi \otimes \psi$ can be computed as follows. Given $x = (x_{i,j}) \in R^{a \times b} = A \otimes B$, we first apply ψ to each of the rows $x_{i,\cdot} \in B$. This yields a new array $y = (y_{i,j}) \in R^{a \times n} = A \otimes N$ with $y_{i,\cdot} = \psi(x_{i,\cdot})$ for all i. We next apply φ to each of the columns $y_{\cdot,j} \in A$. This yields an array $z = (z_{i,j}) \in R^{m \times n} = M \otimes N$ with $z_{\cdot,j} = \varphi(y_{\cdot,j})$ for all j. We claim that $z = (\varphi \circ \psi)(x)$. Indeed, if $x = u \otimes v$, then $y = u \otimes \psi(v)$ and $z = \varphi(u) \otimes \psi(v)$, whence the claim follows by linearity.

Given $x \in A \otimes B$, the above algorithm then allows us to compute $(\varphi \otimes \psi)(x)$ in time

$$\mathsf{C}(\varphi \otimes \psi) \;\leqslant\; a\,\mathsf{C}(\psi) + n\,\mathsf{C}(\varphi) + O(a\,n \lg \min(a, n)\,\mathsf{s}_R + m\,n \lg \min(m, n)\,\mathsf{s}_R). \tag{3.1}$$

More generally, given d linear mappings $\varphi_1 : R^{a_1} \to R^{b_1}, \ldots, \varphi_d : R^{a_d} \to R^{b_d}$, a similar complexity analysis as in the case of the Cooley–Tukey algorithm yields

$$\mathsf{C}(\varphi_1 \otimes \cdots \otimes \varphi_d) \leqslant n_1 \cdots n_d \sum_{i=1}^{d} \frac{\mathsf{C}(\varphi_i)}{n_i} + O(n_1 \cdots n_d \lg (n_1 \cdots n_d)\, \mathsf{s}_R), \tag{3.2}$$

where $n_i = \max(a_i, b_i)$ for $i = 1, \ldots, d$.

3.2. Vector notation and multivariate polynomials

For working with multivariate polynomials, it is convenient to introduce a few notations. Let $\boldsymbol{i} = (i_1, \ldots, i_d)$ and $\boldsymbol{n} = (n_1, \ldots, n_d)$ be d-tuples in $\mathbb{N}^d$. We define

$$\begin{aligned} R^{\boldsymbol{n}} &:= R^{n_1 \cdots n_d} \cong R^{n_1} \otimes \cdots \otimes R^{n_d} \\ |\boldsymbol{n}| &:= n_1 \cdots n_d \\ \mathbb{N}_{\boldsymbol{n}} &:= \{\boldsymbol{i} \in \mathbb{N}^d : i_1 < n_1, \ldots, i_d < n_d\}. \end{aligned}$$

Notice that $\dim R^{\boldsymbol{n}} = |\boldsymbol{n}|$. Elements in $R^{\boldsymbol{n}}$ can be regarded as $n_1 \times \cdots \times n_d$ arrays. On a Turing tape, we recursively represent them as vectors of size n_d with entries in $R^{n_1} \otimes \cdots \otimes R^{n_{d-1}}$.

Given a d-tuple of indeterminates $\boldsymbol{x} = (x_1, \ldots, x_d)$, we define

$$R[\boldsymbol{x}]_{\boldsymbol{n}} := R[x_1]_{n_1} \cdots [x_d]_{n_d} \cong R[x_1]_{n_1} \otimes \cdots \otimes R[x_d]_{n_d} \tag{3.3}$$

$$R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}} := R[x_1]^{\circ}_{n_1} \cdots [x_d]^{\circ}_{n_d} \cong R[x_1]^{\circ}_{n_1} \otimes \cdots \otimes R[x_d]^{\circ}_{n_d}. \tag{3.4}$$

The first tensor product is understood in the sense of R-modules and we regard $R[\boldsymbol{x}]_{\boldsymbol{n}}$ as a subset of $R[\boldsymbol{x}]$. The second tensor product is understood in the sense of R-algebras. Any polynomial A in $R[\boldsymbol{x}]_{\boldsymbol{n}}$ (or cyclic polynomial in $R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}}$) can uniquely be written as a sum

$$A = \sum_{\boldsymbol{i} \in \mathbb{N}_{\boldsymbol{n}}} a_{\boldsymbol{i}} x_1^{i_1} \cdots x_d^{i_d},$$

where $(a_{\boldsymbol{i}})_{\boldsymbol{i} \in \mathbb{N}_{\boldsymbol{n}}} \in R^{\boldsymbol{n}}$ is the vector of coefficients of A. We write $\mathsf{M}_R(\boldsymbol{n})$ for the complexity of multiplying two polynomials in $R[\boldsymbol{x}]_{\boldsymbol{n}}$ and also define $\mathsf{M}^{\circ}_R(\boldsymbol{n}) := \mathsf{m}_{R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}}}$.

3.3. Multivariate Fourier transforms

Assume now that $\boldsymbol{\omega} \in R^d$ is such that ω_i is a principal n_i-th root of unity for $i = 1, \ldots, d$ and invertible positive integers $n_1, \ldots, n_d$. As in the univariate case, cyclic polynomials $A \in R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}}$ can naturally be evaluated at points of the form $(\omega_1^{i_1}, \ldots, \omega_d^{i_d})$. We now define the discrete Fourier transform of A to be the vector $\mathrm{DFT}_{\boldsymbol{\omega}}(A) \in R^{\boldsymbol{n}}$ with

$$\mathrm{DFT}_{\boldsymbol{\omega}}(A)_{\boldsymbol{i}} := A(\omega_1^{i_1}, \ldots, \omega_d^{i_d})$$

for all $\boldsymbol{i} \in \mathbb{N}_{\boldsymbol{n}}$. Multivariate Fourier transforms again provide us with an isomorphism between $R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}}$ and $R^{\boldsymbol{n}}$. Writing $\mathsf{F}_R(\boldsymbol{n})$ for the cost to compute such a transform with respect to a most suitable $\boldsymbol{\omega}$, we thus have $\mathsf{C}(R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}} \leftrightarrow R^{\boldsymbol{n}}) \leqslant \mathsf{F}_R(\boldsymbol{n}) + |\boldsymbol{n}|\, \mathsf{d}_{R,\mathbb{Z}} + O(|\boldsymbol{n}|\, \mathsf{s}_R)$.

We claim that multivariate Fourier transforms can be regarded as the tensor product of univariate Fourier transforms with respect to each of the variables, that is,

$$\mathrm{DFT}_{\boldsymbol{\omega}} = \mathrm{DFT}_{\omega_1} \otimes \cdots \otimes \mathrm{DFT}_{\omega_d}. \tag{3.5}$$

Indeed, given a vector $a \in R^{\boldsymbol{n}}$ of the form $a = a_1 \otimes \cdots \otimes a_d$ with $a_k \in R^{n_k}$ for each k, we have

$$\mathrm{DFT}_{\boldsymbol{\omega}}(a)_{\boldsymbol{i}} = A(\omega_1^{i_1}, \ldots, \omega_d^{i_d}) = A_1(\omega_1^{i_1}) \cdots A_d(\omega_d^{i_d}), \tag{3.6}$$

where $A_k = (a_k)_0 + \cdots + (a_k)_{n_k-1} x_k^{n_k-1}$ for each k, and A is as above. The relation (3.5) follows from (3.6) by linearity, since the a of the form $a_1 \otimes \cdots \otimes a_d$ span $R^{\boldsymbol{n}}$.

In particular, in order to compute a multivariate Fourier transform, it suffices to compute univariate Fourier transforms with respect to each of the variables, as described in section 3.1. It follows that

$$\mathsf{F}_R(\boldsymbol{n}) \leqslant |\boldsymbol{n}| \sum_{i=1}^{d} \frac{\mathsf{F}_R(n_i)}{n_i} + O(|\boldsymbol{n}| \lg |\boldsymbol{n}| \mathsf{s}_R). \tag{3.7}$$

The technique of FFT-multiplication from section 2.2 also naturally generalizes to multivariate polynomials in $R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}}$. This yields the bound

$$\mathsf{M}^{\circ}_R(\boldsymbol{n}) \leqslant 3\mathsf{F}_R(\boldsymbol{n}) + |\boldsymbol{n}| (\mathsf{m}_R + \mathsf{d}_{R,\mathbb{Z}}) + O(|\boldsymbol{n}| \mathsf{s}_R). \tag{3.8}$$

If $R = C[u]^{\ominus}_N$, as in section 2.4, and $n_1, \ldots, n_d$ all divide $2N$, then (3.7) and (3.8) become simply

$$\mathsf{F}_R(\boldsymbol{n}) \leqslant O(|\boldsymbol{n}| \lg |\boldsymbol{n}| \mathsf{s}_R) \tag{3.9}$$

$$\mathsf{M}^{\circ}_R(\boldsymbol{n}) \leqslant |\boldsymbol{n}| (\mathsf{m}_R + \mathsf{d}_{R,\mathbb{Z}}) + O(|\boldsymbol{n}| \lg |\boldsymbol{n}| \mathsf{s}_R). \tag{3.10}$$

3.4. CRT transforms

Let $\boldsymbol{n} = (n_1, \ldots, n_d)$ be a tuple of pairwise coprime numbers and $n = n_1 \cdots n_d$. The Chinese remainder theorem provides us with an isomorphism of abelian groups

$$\begin{aligned} \mathbb{Z}/(n_1 \cdots n_d \mathbb{Z}) &\xrightarrow{\sigma} \mathbb{Z}/(n_1\mathbb{Z}) \times \cdots \times \mathbb{Z}/(n_d\mathbb{Z}) \\ i &\longmapsto (\sigma_1(i), \ldots, \sigma_d(i)). \end{aligned}$$

Any such isomorphism induces an isomorphism of R-algebras

$$\begin{aligned} R[x]^{\circ}_{n_1 \cdots n_d} &\xrightarrow{\varrho} R[x_1]^{\circ}_{n_1} \otimes \cdots \otimes R[x_d]^{\circ}_{n_d} \cong R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}} \\ x^i &\longmapsto x_1^{\sigma_1(i)} \cdots x_d^{\sigma_d(i)} \end{aligned}$$

that boils down to a permutation of coefficients when using the usual power bases for $R[x]^{\circ}_{n_1 \cdots n_d}$ and $R[x_1]^{\circ}_{n_1} \otimes \cdots \otimes R[x_d]^{\circ}_{n_d}$. For a suitable choice of σ, we show in section A.3 of the appendix that this permutation can be computed efficiently on Turing machines, which yields

$$\mathsf{C}(R[x]^{\circ}_n \leftrightarrow R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}}) = O(n \lg n \mathsf{s}_R).$$

We will also need a multivariate generalization of this formula: let $\boldsymbol{p} = (p_1, \ldots, p_d)$, $\boldsymbol{q} = (q_1, \ldots, q_d)$ and $\boldsymbol{n} = (p_1 q_1, \ldots, p_d q_d)$ be such that p_i and q_i are coprime for $i = 1, \ldots, d$. Then

$$\mathsf{C}(R[\boldsymbol{x}]^{\circ}_{\boldsymbol{n}} \leftrightarrow R[\boldsymbol{y}]^{\circ}_{\boldsymbol{p}}[\boldsymbol{z}]^{\circ}_{\boldsymbol{q}}) = O(|\boldsymbol{n}| \lg |\boldsymbol{n}| \mathsf{s}_R).$$

This is a result of Corollary A.8 in the appendix.

3.5. Negacyclic CRT transforms

Assume now that $d = 2$ and let $n = n_1 n_2$, where n_1 and n_2 are coprime and n_2 is odd. Then we have isomorphisms of R-algebras

$$\begin{aligned} R[x]^{\ominus}_n &\cong R[x]^{\circ}_{2n}/(x^n + 1) \\ R[x_1]^{\ominus}_{n_1} &\cong R[x_1]^{\circ}_{2n_1}/(x_1^{n_1} + 1). \end{aligned}$$

Given $c = (2n_1)^{-1} \bmod n_2$, the map $i \mapsto (i \bmod 2n_1, c\,i \bmod n_2)$ determines an isomorphism of abelian groups between $\mathbb{Z}/(2n\,\mathbb{Z})$ and $\mathbb{Z}/(2n_1\,\mathbb{Z}) \times \mathbb{Z}/(n_2\,\mathbb{Z})$. As explained in the previous subsection, this gives rise to an isomorphism of R-algebras

$$\begin{aligned} R[x]^{\circ}_{2n} &\xrightarrow{\varrho} R[x_1]^{\circ}_{2n_1}[x_2]^{\circ}_{n_2} \\ x^i &\longmapsto x_1^i x_2^{ci}. \end{aligned}$$

In view of Lemma A.4, this isomorphism can be computed efficiently. Moreover, $\varrho(x^n) = x_1^{n_1 n_2} x_2^{c n_1 n_2} = (x_1^{n_1})^{n_2} (x_2^{n_2})^{c n_1} = x_1^{n_1}$, since n_2 is odd and $x_2^{n_2} = 1$. This yields a natural isomorphism $\bar{\varrho}: R[x]^{\ominus}_{n} \to R[x_1]^{\ominus}_{n_1}[x_2]^{\circ}_{n_2}$ of R-algebras such that the following diagram commutes:

$$\begin{array}{ccc} R[x]^{\circ}_{2n} & \xrightarrow{\varrho} & R[x_1]^{\circ}_{2n_1}[x_2]^{\circ}_{n_2} \\ \downarrow & & \downarrow \\ R[x]^{\ominus}_{n} & \xrightarrow{\bar{\varrho}} & R[x_1]^{\ominus}_{n_1}[x_2]^{\circ}_{n_2} \end{array}$$

The projection $R[x]^{\circ}_{2n} \to R[x]^{\ominus}_{n}$ admits an obvious right inverse that sends a polynomial $a_0 + \cdots + a_{n-1} x^{n-1}$ modulo $x^n + 1$ to $a_0 + \cdots + a_{n-1} x^{n-1}$. This lift can clearly be computed in linear time, so we again obtain

$$\mathsf{C}(R[x]^{\ominus}_{n} \leftrightarrow R[x_1]^{\ominus}_{n_1}[x_2]^{\circ}_{n_2}) \;=\; O(n \lg n\, \mathsf{s}_R).$$

Alternatively, one may compute $\bar{\varrho}$ in a similar way as the isomorphism of R-algebras between $R[x]^{\circ}_{n} \to R[x_1]^{\circ}_{n_1}[x_2]^{\circ}_{n_2}$ from the previous subsection, while inverting signs whenever appropriate.

3.6. Tricyclic CRT transforms

In section 8, we will also need a triadic variant of the negacyclic CRT transforms from the previous subsection. More precisely, assume that $n = n_1 n_2$, where n_1 and n_2 are coprime, n_1 is even, and n_2 is not divisible by 3. Then we have isomorphisms of R-algebras

$$\begin{aligned} R[x]^{\varnothing}_{n} &\cong R[x]^{\circ}_{3n/2} / (x^n + x^{n/2} + 1) \\ R[x_1]^{\varnothing}_{n_1} &\cong R[x_1]^{\circ}_{3n_1/2} / (x_1^{n_1} + x_1^{n_1/2} + 1). \end{aligned}$$

Given $c = (3n_1/2)^{-1} \bmod n_2$, the map $i \mapsto (i \bmod (3n_1/2), c\,i \bmod n_2)$ determines an isomorphism of abelian groups between $\mathbb{Z}/((3n/2)\,\mathbb{Z})$ and $\mathbb{Z}/((3n_1/2)\,\mathbb{Z}) \times \mathbb{Z}/(n_2\,\mathbb{Z})$. In a similar way as in the previous subsection, this leads to an isomorphism of R-algebras

$$\begin{aligned} R[x]^{\circ}_{3n/2} &\xrightarrow{\varrho} R[x_1]^{\circ}_{3n_1/2}[x_2]^{\circ}_{n_2} \\ x^i &\longmapsto x_1^i x_2^{ci}, \end{aligned}$$

and a commutative diagram

$$\begin{array}{ccc} R[x]^{\circ}_{3n/2} & \xrightarrow{\varrho} & R[x_1]^{\circ}_{3n_1/2}[x_2]^{\circ}_{n_2} \\ \downarrow & & \downarrow \\ R[x]^{\varnothing}_{n} & \xrightarrow{\bar{\varrho}} & R[x_1]^{\varnothing}_{n_1}[x_2]^{\circ}_{n_2} \end{array}$$

in which all maps and their right inverses can be computed efficiently. This again yields

$$\mathsf{C}(R[x]^{\varnothing}_{n} \leftrightarrow R[x_1]^{\varnothing}_{n_1}[x_2]^{\circ}_{n_2}) \;=\; O(n \lg n\, \mathsf{s}_R).$$

We will qualify this kind of conversions between $R[x]^{\varnothing}_{n}$ and $R[x_1]^{\varnothing}_{n_1}[x_2]^{\circ}_{n_2}$ as *tricyclic CRT transforms*.

4. RADER REDUCTION

4.1. Univariate Rader reduction

Let R be a ring and let $\omega \in R$ be a principal p-th root of unity for some prime number p. The multiplicative group $\mathbb{F}_p^*$ is cyclic, of order $p-1$. Let $g \in \{1, \ldots, p-1\}$ be such that $g \bmod p$ is a generator of this cyclic group. Given a polynomial $A = a_0 + \cdots + a_{p-1}x^{p-1}$ and $j \in \{1, \ldots, p-1\}$ with $j = g^l \operatorname{rem} p$, we have

$$\begin{aligned}
\mathrm{DFT}_\omega(a)_{g^l \operatorname{rem} p} &= a_0 + \sum_{i=1}^{p-1} a_i \omega^{i(g^l \operatorname{rem} p)} \\
&= a_0 + \sum_{k=0}^{p-2} a_{g^{-k} \operatorname{rem} p} \omega^{(g^{-k} \operatorname{rem} p)(g^l \operatorname{rem} p)} \\
&= a_0 + \sum_{k=0}^{p-2} a_{g^{-k} \operatorname{rem} p} \omega^{g^{l-k} \operatorname{rem} p}.
\end{aligned}$$

Setting

$$\begin{aligned}
u_k &= a_{g^{-k} \operatorname{rem} p} - a_0 & U &= u_0 + \cdots + u_{p-2} y^{p-2} \\
v_l &= \omega^{g^l \operatorname{rem} p} & V &= v_0 + \cdots + v_{p-2} y^{p-2} \\
w_l &= \mathrm{DFT}_\omega(a)_{g^l \operatorname{rem} p} & W &= w_0 + \cdots + w_{p-2} y^{p-2},
\end{aligned}$$

it follows that $W = UV$, when regarding U, V and W as elements of $R[y]^\circ_{p-1}$, since

$$V \left[\sum_{k=0}^{p-2} y^k \right] = \sum_{k=0}^{p-2} \left[\sum_{l=0}^{p-2} \omega^{g^l \operatorname{rem} p} \right] y^k = \sum_{k=0}^{p-2} \left[\sum_{l=1}^{p-1} \omega^l \right] y^k = -\sum_{k=0}^{p-2} y^k.$$

Notice that the vector V can be precomputed since it does not depend on a.

In summary, we have reduced the computation of a discrete Fourier transform of length p to a cyclic convolution of length $p-1$ and $O(p)$ additions in R. This reduction is due to Rader [46], except for the minor twist that we subtract a_0 in the definition of u_k. As far as we are aware, this twist is essentially due to Nussbaumer [43, section 5.2.2]. In the next subsection, it allows us to compute Rader transforms without increasing the dimension from p to $p+1$.

4.2. Univariate Rader transforms

Let us now show how to present Rader reduction using suitable linear transforms, called left, right and inverse Rader transforms (LRT, RRT and IRT). These transforms depend on the choice of g; from now on, we assume that g has been fixed once and for all. Given $a \in R^p$ and $A = a_0 + \cdots + a_{p-1}x^{p-1}$, let U, V and W be as in section 4.1. Setting

$$R[y]^\sharp_p = R \oplus R[y]^\circ_{p-1},$$

we first introduce the distinguished element $\mathrm{RRT}_p(\omega) \in R[y]^\sharp_p$ by

$$\mathrm{RRT}_p(\omega) = (1, V).$$

We also define $\mathrm{LRT}_p(a) \in R[y]^\sharp_p$ by

$$\mathrm{LRT}_p(a) = (A(1), U),$$

so that $\mathrm{LRT}_p : R^p \to R[y]_p^\sharp$ is a linear mapping. We can retrieve $\mathrm{DFT}_\omega(a)$ from the product $(A(1), W) = \mathrm{LRT}_p(a)\,\mathrm{RRT}_p(\omega)$ using

$$\begin{aligned} \mathrm{DFT}_\omega(a)_0 &= A(1) \\ \mathrm{DFT}_\omega(a)_{g^l \bmod p} &= w_l. \end{aligned}$$

We denote by $\mathrm{IRT}_p : R[y]_p^\sharp \to R^p$ the linear mapping with $\mathrm{IRT}_p(A(1), W) = \mathrm{DFT}_\omega(a)$. Altogether, we have

$$\mathrm{DFT}_\omega(a) = \mathrm{IRT}_p(\mathrm{LRT}_p(a)\,\mathrm{RRT}_p(\omega)).$$

The linear mappings LRT_p and IRT_p boil down to permutations and a linear number of additions and subtractions. Now permutations can be computed efficiently on Turing machines, as recalled in section A.1 of the appendix. More precisely, since ω is a principal root of unity of order p, we necessarily have $|R| \geqslant p$. Then the permutation of p coefficients in R of bitsize $\mathsf{s}_R \geqslant \lg p$ can be done using a merge sort in time $O(p \lg p\, \mathsf{s}_R)$, whence

$$\begin{aligned} \mathsf{C}(\mathrm{LRT}_p) &\leqslant O(p \lg p\, \mathsf{s}_R) \\ \mathsf{C}(\mathrm{IRT}_p) &\leqslant O(p \lg p\, \mathsf{s}_R). \end{aligned}$$

It follows that

$$\mathsf{F}_R(p) \leqslant \mathsf{M}_R^\circ(p-1) + \mathsf{M}_R^\circ(1) + O(p \lg p\, \mathsf{s}_R). \tag{4.1}$$

In this bound, the cost $\mathsf{M}_R^\circ(1)$ corresponds to the trivial multiplication $1 \cdot A(1)$. We included this term in anticipation of the multivariate generalization (4.3) below. We also recall that we assumed V to be precomputed. This precomputation can be done in time $O(p\, \mathsf{m}_R + p \lg p\, \mathsf{s}_R)$: we first compute the powers ω^i for $i = 0, \ldots, p-2$ and then permute them appropriately.

4.3. Multivariate Rader transforms

Assume now that $p_1, \ldots, p_d$ are prime numbers, that we have fixed generators $g_1, \ldots, g_d$ for the cyclic groups $\mathbb{F}_{p_1}^*, \ldots, \mathbb{F}_{p_d}^*$, and that $\omega_1, \ldots, \omega_d \in R$ are such that ω_i is a principal p_i-th root of unity for $i = 1, \ldots, d$. Let $\boldsymbol{p} = (p_1, \ldots, p_d)$, $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_d)$, $\boldsymbol{x} = (x_1, \ldots, x_d)$ and $\boldsymbol{y} = (y_1, \ldots, y_d)$. We define

$$\begin{aligned} R[\boldsymbol{y}]_{\boldsymbol{p}}^\sharp &:= R[y_1]_{p_1}^\sharp \cdots [y_d]_{p_d}^\sharp \\ &\cong R[y_1]_{p_1}^\sharp \otimes \cdots \otimes R[y_d]_{p_d}^\sharp. \end{aligned}$$

Using the distributivity of $\otimes$ with respect to $\oplus$, the algebra $R[\boldsymbol{y}]_{\boldsymbol{p}}$ is a direct sum of algebras of cyclic multivariate polynomials. More precisely, let $\mathcal{W} = \{0,1\}^d$. For each index $w \in \mathcal{W}$, let $\boldsymbol{r} = \boldsymbol{r}_w \in \mathbb{N}^d$ be such that $r_k = 1$ if $w_k = 0$ and $r_k = p_k - 1$ if $w_k = 1$ for $k = 1, \ldots, d$. Then

$$R[\boldsymbol{y}]_{\boldsymbol{p}}^\sharp \cong \bigoplus_{w \in \mathcal{W}} R[\boldsymbol{y}]_{\boldsymbol{r}_w}^\circ.$$

Notice that

$$\dim R[\boldsymbol{y}]_{\boldsymbol{p}}^\sharp = \sum_{w \in \mathcal{W}} |\boldsymbol{r}_w| = |\boldsymbol{p}|.$$

Elements in $\bigoplus_{w\in\mathcal{W}} R[\boldsymbol{y}]^{\circ}_{r_w}$ correspond to sums $x=\sum_{w\in\mathcal{W}} x_w$ with $x_i\in R[\boldsymbol{y}]^{\circ}_{r_w}$. Assuming the lexicographical ordering on $\mathcal{W}$, such sums are represented on a Turing tape by concatenating the representations of the components x_w: see section A.5 in the appendix for details. We have

$$\mathsf{C}\left(R[\boldsymbol{y}]^{\sharp}_{\boldsymbol{p}} \leftrightarrow \bigoplus_{w\in\mathcal{W}} R[\boldsymbol{y}]^{\circ}_{r_w}\right) \;=\; O(|\boldsymbol{p}|\, d\, \mathsf{s}_R), \tag{4.2}$$

by Lemma A.9 (we may take $\delta=2$).

The tensor product of the mappings $\mathrm{LRT}_{p_i}\colon R^{p_i}\to R[y_i]^{\sharp}_{p_i}$ provides us with a mapping

$$\mathrm{LRT}_{\boldsymbol{p}} \;:=\; \mathrm{LRT}_{p_1}\otimes\cdots\otimes\mathrm{LRT}_{p_d}\colon R^{\boldsymbol{p}}\to R[\boldsymbol{y}]^{\sharp}_{\boldsymbol{p}}.$$

Similarly, we obtain a mapping

$$\mathrm{IRT}_{\boldsymbol{p}} \;:=\; \mathrm{IRT}_{p_1}\otimes\cdots\otimes\mathrm{IRT}_{p_d}\colon R[\boldsymbol{y}]^{\sharp}_{\boldsymbol{p}}\to R^{\boldsymbol{p}}$$

and a distinguished element

$$\mathrm{RRT}_{\boldsymbol{p}}(\boldsymbol{\omega}) \;:=\; \mathrm{RRT}_{p_1}(\omega_1)\otimes\cdots\otimes\mathrm{RRT}_{p_d}(\omega_d)\;\in R[\boldsymbol{y}]^{\sharp}_{\boldsymbol{p}}.$$

For any $a_1\in R^{p_1},\ldots,a_d\in R^{p_d}$ and $\boldsymbol{a}=a_1\otimes\cdots\otimes a_d$, we now have

$$\begin{aligned}
\mathrm{DFT}_{\boldsymbol{\omega}}(\boldsymbol{a}) &= \mathrm{DFT}_{\omega_1}(a_1)\otimes\cdots\otimes\mathrm{DFT}_{\omega_d}(a_d)\\
&= \mathrm{IRT}_{p_1}(\mathrm{LRT}_{p_1}(a_1)\,\mathrm{RRT}_{p_1}(\omega_1))\otimes\cdots\otimes\mathrm{IRT}_{p_d}(\mathrm{LRT}_{p_d}(a_d)\,\mathrm{RRT}_{p_d}(\omega_d))\\
&= \mathrm{IRT}_{\boldsymbol{p}}(\mathrm{LRT}_{p_1}(a_1)\,\mathrm{RRT}_{p_1}(\omega_1)\otimes\cdots\otimes\mathrm{LRT}_{p_d}(a_d)\,\mathrm{RRT}_{p_d}(\omega_d))\\
&= \mathrm{IRT}_{\boldsymbol{p}}((\mathrm{LRT}_{p_1}(a_1)\otimes\cdots\otimes\mathrm{LRT}_{p_d}(a_d))\,\mathrm{RRT}_{\boldsymbol{p}}(\boldsymbol{\omega}))\\
&= \mathrm{IRT}_{\boldsymbol{p}}(\mathrm{LRT}_{\boldsymbol{p}}(\boldsymbol{a})\,\mathrm{RRT}_{\boldsymbol{p}}(\boldsymbol{\omega})).
\end{aligned}$$

By linearity, it follows that

$$\mathrm{DFT}_{\boldsymbol{\omega}}(\boldsymbol{a}) \;=\; \mathrm{IRT}_{\boldsymbol{p}}(\mathrm{LRT}_{\boldsymbol{p}}(\boldsymbol{a})\,\mathrm{RRT}_{\boldsymbol{p}}(\boldsymbol{\omega}))$$

for all $\boldsymbol{a}\in R^{\boldsymbol{p}}$.

From the complexity point of view, the relation (3.2) implies

$$\begin{aligned}
\mathsf{C}(\mathrm{LRT}_{\boldsymbol{p}}) &\leqslant |\boldsymbol{p}|\sum_{i=1}^{d}\frac{\mathsf{C}(\mathrm{LRT}_{p_i})}{p_i} + O(|\boldsymbol{p}|\lg|\boldsymbol{p}|\,\mathsf{s}_R) \leqslant O(|\boldsymbol{p}|\lg|\boldsymbol{p}|\,\mathsf{s}_R)\\
\mathsf{C}(\mathrm{IRT}_{\boldsymbol{p}}) &\leqslant |\boldsymbol{p}|\sum_{i=1}^{d}\frac{\mathsf{C}(\mathrm{IRT}_{p_i})}{p_i} + O(|\boldsymbol{p}|\lg|\boldsymbol{p}|\,\mathsf{s}_R) \leqslant O(|\boldsymbol{p}|\lg|\boldsymbol{p}|\,\mathsf{s}_R).
\end{aligned}$$

Using (4.2) and $d\leqslant\lg|\boldsymbol{p}|$, this leads to the following multivariate analogue of (4.1):

$$\mathsf{F}_R(\boldsymbol{p}) \;\leqslant\; \sum_{w\in\mathcal{W}} \mathsf{M}^{\circ}_R(\boldsymbol{r}_w) + O(|\boldsymbol{p}|\lg|\boldsymbol{p}|\,\mathsf{s}_R). \tag{4.3}$$

As in the case of (4.1), this bound does not include the cost of the precomputation of $\mathrm{RRT}_{\boldsymbol{p}}(\boldsymbol{\omega})$. This precomputation can be performed in time $O(|\boldsymbol{p}|\,\mathsf{m}_R+|\boldsymbol{p}|\lg|\boldsymbol{p}|\,\mathsf{s}_R)$ using the same "list and sort" technique as in the univariate case.

5. PRIME NUMBERS IN ARITHMETIC PROGRESSIONS

5.1. The smallest prime number in an arithmetic progression

Let $\mathbb{P}$ be the set of all prime numbers. Given two integers $k>0$ and $0<a<k$ with $\gcd(a,k)=1$, we define $P(a,k)\in\mathbb{P}$ and $P(k)\in\mathbb{P}$ by

$$\begin{aligned}
P(a,k) &:= \min\{ck+a : c\in\mathbb{N}, ck+a\in\mathbb{P}\}\\
P(k) &:= \max\{P(a,k) : 0<a<k, \gcd(a,k)=1\}.
\end{aligned}$$

A constant $L > 1$ with the property that $P(k) = O(k^L)$ is called a *Linnik constant* (notice that we allow the implicit constants k_0 and C with $P(k) \leqslant C k^L$ for $k \geqslant k_0$ to depend on L).

The existence of a Linnik constant $L > 1$ was shown by Linnik [39, 40]. The smallest currently known value $L = 5.18$ is due to Xylouris [57]. Assuming the Generalized Riemann Hypothesis (GRH), it is known that any $L > 2$ is a Linnik constant [28]. Unfortunately, these bounds are still too weak for our purposes.

Both heuristic probabilistic arguments and numerical evidence indicate that $P(k)$ satisfies the much sharper bound $P(k) = O(\varphi(k) \log^2 k) = O(k \log^2 k)$, where φ stands for Euler's totient function. More precisely, based on such evidence [38], Li, Pratt and Shakan conjecture that

$$\liminf_k \frac{P(k)}{\varphi(k) \log^2 k} = 1, \qquad \limsup_k \frac{P(k)}{\varphi(k) \log^2 k} = 2.$$

Unfortunately, a proof of this conjecture currently seems to be out of reach. At any rate, this conjecture is much stronger than the assumption that $L < 1 + 2^{-1162}$ in Theorem 1.2.

5.2. Multiple prime numbers in arithmetic progressions

Given integers $k > 0$, $\nu > 0$, and $a > 0$ with $\gcd(a,k) = 1$, let $P_\nu(a,k)$ denote the ν-th prime number in the arithmetic progression $a, k+a, 2k+a, \ldots$, so that $P(a,k) = P_1(a,k)$. Also define $P_\nu(k) := \max\{P_\nu(a,k) : 0 < a < k, \gcd(a,k) = 1\}$.

LEMMA 5.1. *Let L be a Linnik constant. Then for any fixed integer $\nu > 0$, we have*

$$P_\nu(k) = O((k \log k)^L).$$

Remark. Note that the implied big-Oh constant depends on ν; a similar remark applies to Lemma 5.4 below.

Proof. Let $C > 0$ be an absolute constant with $P(\ell) < C\,\ell^L$ for all ℓ. Let $k > 0$ be given and let $a \in \{1, \ldots, k-1\}$ be a fixed remainder class with $\gcd(a,k) = 1$.

Let q be the smallest prime number such that $q > \nu$ and $q \nmid k$. The prime number theorem implies that $q = O(\log k)$. For each $i \in \{1, \ldots, \nu\}$, let $r_i \in \{0, \ldots, kq-1\}$ be such that $r_i \equiv i \bmod q$ and $r_i \equiv a \bmod k$. Such an integer exists since q and k are coprime. Notice also that r_i and kq are coprime. Now consider the prime numbers $p_i = P(r_i, kq)$ for $i = 1, \ldots, \nu$. By construction, we have $p_i \equiv r_i \equiv i \bmod q$, so the numbers p_i are pairwise distinct. We also have $p_i \equiv a \bmod k$ and $p_i \leqslant C\,(kq)^L$, whence $P_\nu(a,k) \leqslant \max_{1 \leqslant i \leqslant \nu} p_i \leqslant C\,(kq)^L = O((k \log k)^L)$. □

Remark 5.2. For numbers k with a bounded number of prime factors, we actually have $q = O(1)$ in the proof of Lemma 5.1, which leads to the sharper bound $P_\nu(k) = O(k^L)$. It is possible that the bound $P_\nu(k) = O(k^L)$ always holds, but we were unable to prove this so far. If ν is no longer fixed, then it is not hard to see that $q = O(\nu + \log k)$, which yields $P_\nu(k) = O((k\,(\nu + \log k))^L)$. Concerning a generalization of the Li–Pratt–Shakan conjecture, it is likely that there exist constants a_ν and b_ν with

$$\liminf_k \frac{P_\nu(k)}{\varphi(k) \log^2 k} = a_\nu, \qquad \limsup_k \frac{P_\nu(k)}{\varphi(k) \log^2 k} = b_\nu.$$

It is actually even likely that $a_\nu = \nu$ and $b_\nu = 2\nu$. At any rate, it would be interesting to empirically determine a_ν and b_ν *via* numerical experiments.

5.3. Forcing coprime multipliers

When searching for multiple prime numbers $q_1 k + a, q_2 k + a, \ldots, q_\nu k + a$ in an arithmetic progression, as in the previous subsection, it will be useful in sections 7 and 8 to insist that the multipliers $q_1, \ldots, q_\nu$ be pairwise coprime. The simplest way to force this is to define the q_i by induction for $i = 1, 2, \ldots$ using

$$q_i := \min \{q' \geqslant 2 : q' k + a \in \mathbb{P}, \gcd(q', q_1 \cdots q_{i-1}) = 1\}. \tag{5.1}$$

(In Lemma 5.3 below, we will show that such a value of q' exists for all i, provided that k is even.) We then define $P^*_\nu(a,k) := q_\nu k + a$ and $P^*_\nu(k) := \max\{P^*_\nu(a,k) : 0 < a < k, \gcd(a,k) = 1\}$.

LEMMA 5.3. *Let $L > 1$ be a Linnik constant and assume that k is even. Given an integer $q \geqslant 1$, there exists an integer $q' \geqslant 2$ such that $q' k + a$ is prime,* $\gcd(q', q) = 1$, *and*

$$q' = O(q^L k^{L-1}).$$

Proof. Modulo the replacement of q by $5q$, we may assume without loss of generality that $5 \mid q$. First we show that for each prime $p \mid q$, there exists an integer $r_p \in \{p-2, p-1\}$ such that $p \nmid r_p$ and $p \nmid r_p k + a$. Indeed, if $p \mid k$ then $p \nmid a$ so we may take $r_p = p - 1$; if $p \nmid k$, then $p \geqslant 3$, so at least one $r_p \in \{p-2, p-1\}$ satisfies $r_p \not\equiv 0 \bmod p$ and $r_p \not\equiv -a/k \bmod p$. Now, using the Chinese remainder theorem, we may construct an integer r such that $0 \leqslant r < \prod_{p \mid q} p \leqslant q$ and $r \equiv r_p \bmod p$ for all $p \mid q$. Then we have $\gcd(r, q) = 1$ and $\gcd(rk + a, q) = 1$. Since $r_5 \equiv 3 \bmod 5$ or $r_5 \equiv 4 \bmod 5$, we also must have $r \geqslant 2$.

Now $\gcd(rk + a, qk) = 1$, since $\gcd(rk + a, q) = 1$ and $\gcd(rk + a, k) = \gcd(a, k) = 1$. It follows that $P(rk + a, qk)$ is well defined. Let $c \geqslant 0$ be the integer with $P(rk + a, qk) = cqk + rk + a = (cq + r)k + a$ and take $q' = cq + r \geqslant 2$, so that $\gcd(q', q) = \gcd(r, q) = 1$. We conclude by observing that $q'k + a = P(rk + a, qk) = O((qk)^L)$, whence $q' = O(q^L k^{L-1})$. □

LEMMA 5.4. *Let L be a Linnik constant and assume that k is even. Then for any fixed integer $\nu > 0$, we have*

$$P^*_\nu(k) = O(k^{1 + (L+1)^{\nu-1}(L-1)}).$$

Proof. Let us prove by induction on i that q_i defined as in (5.1) satisfies

$$q_i = O(k^{(L+1)^{i-1}(L-1)}).$$

Indeed, setting $q = q_1 \cdots q_{i-1}$, the induction hypothesis implies

$$q = O(k^{((L+1)^{i-2} + (L+1)^{i-3} + \cdots + 1)(L-1)}) = O(k^{((L+1)^{i-1} - 1)(L-1)/L}).$$

Lemma 5.3 then yields

$$q_i = O(q^L k^{L-1}) = O(k^{((L+1)^{i-1} - 1)(L-1)} k^{L-1}) = O(k^{(L+1)^{i-1}(L-1)}),$$

whence the result. □

Remark 5.5. Our bound for $P^*_\nu(k)$ seems overly pessimistic. It is possible that a bound $P^*_\nu(k) = O(k^L)$ or $P^*_\nu(k) = O((k \log k)^L)$ always holds, but we were unable to prove this so far. It would also be interesting to investigate counterparts of the Li–Pratt–Shakan conjecture.

6. PROOF OF THEOREM 1.1

Let $L > 1$ be a Linnik constant with $L < 1 + {}^1\!/_{303}$. Our algorithm proceeds as follows.

Step 0: setting up the parameters. Assume that we wish to multiply two n-bit integers. For sufficiently large $n \geqslant n_0$ and suitable parameters m, s, $p_1, \ldots, p_d$, we will reduce this problem to a multiplication problem in $(\mathbb{Z}/m\mathbb{Z})[z]^{\ominus}_{sp_1 \cdots p_d}$. If $n < n_0$, then any traditional multiplication algorithm can be used. The threshold n_0 should be chosen such that various asymptotic relations encountered in the proof below hold for all $n \geqslant n_0$. The other parameters are chosen as follows:

- $\epsilon := 10^{-6}$.
- $\ell := 2^{\lg(n^\epsilon)}$ is the smallest power of two with $\ell \geqslant n^\epsilon$.
- $p_i := P_i(\ell + 1, 2\ell)$ for $i = 1, 2, \ldots$
- $d := \min\{d \in \mathbb{N} : \ell^3 p_1 \cdots p_d \geqslant n\}$.
- $\pi := P(1, \nu)$, where $\nu := \operatorname{lcm}(\ell^3, p_1 - 1, \ldots, p_d - 1)\, p_1 \cdots p_d$.
- $\lambda := \lceil \lg^2 n / \lg \pi \rceil$ and $m := \pi^\lambda$.
- $s := 2^{\lg(5n/(p_1 \cdots p_d \lg^2 n))}$
- $N := s p_1 \cdots p_d$.
- $b := \lfloor \lg(2m/N)/2 \rfloor$ is maximal with $2^{2b} N/2 \leqslant m$.

We claim that $p_1, \ldots, p_d$ and the rest of these parameters can be computed in linear time $O(n)$. Recall from [2] that we can test whether a β-bit integer is prime in time $O(\beta^{O(1)})$. We clearly have $d \leqslant 1/\epsilon$, whence $p_i \leqslant O((\ell \log \ell)^L)$ for $i = 1, \ldots, d$, by Lemma 5.1. For sufficiently large n, this means that

$$\ell \leqslant p_1 < \cdots < p_d \leqslant \ell^{L+\epsilon}.$$

The brute force computation of $p_1, \ldots, p_d$ requires $(p_d - 1)/\ell \leqslant \ell^{L+\epsilon-1}$ primality tests of total cost $\ell^{L+\epsilon-1} (\log n)^{O(1)} = O(n)$. From $(p_d - 1)/\ell \leqslant \ell^{L+\epsilon-1}$, we also deduce $\operatorname{lcm}(\ell^3, p_1 - 1, \ldots, p_d - 1) \leqslant \ell^{3+d(L+\epsilon-1)} \leqslant O(\ell^3 n^{L-1+\epsilon})$ and $\nu \leqslant O(n^{L+\epsilon} p_d) \leqslant O(n^{L+3\epsilon})$. In particular, $\pi \leqslant O((n^{L+3\epsilon})^L)$; for n sufficiently large, we thus get

$$n \leqslant \pi \leqslant n^{L^2+4\epsilon L}.$$

In order to compute π, we need to perform $O(n^{L^2+4\epsilon L-1}) = O(n^{1/2})$ primality tests of total cost $O(n^{1/2} (\log n)^{O(1)}) = O(n)$. This completes the proof of our claim.

Let us proceed with a few useful bounds. From $\log \pi \asymp \log n$ we get

$$\lg m = \lambda (\lg \pi + O(1)) = \left(\frac{\lg^2 n}{\lg \pi} + O(1) \right) (\lg \pi + O(1)) = \lg^2 n + O(\lg n),$$

which in turn yields

$$b = \frac{1}{2} \lg m + O(\lg N) = \frac{1}{2} \lg m + O(\lg n) = \frac{1}{2} \lg^2 n + O(\lg n).$$

Since $n \leqslant \ell^3 p_1 \cdots p_d$, we also have

$$2s \leqslant \frac{20n}{p_1 \cdots p_d \lg^2 n} \leqslant \ell^3$$

and $2s \mid \ell^3$, for sufficiently large n. Inversely, $\ell^3 p_1 \cdots p_{d-1} < n$ implies

$$\frac{1}{s} \leqslant \frac{p_1 \cdots p_d \lg^2 n}{n} = \frac{\lg^2 n}{\ell^3} \cdot \frac{\ell^3 p_1 \cdots p_d}{n} \leqslant \frac{\lg^2 n}{\ell^3} p_d \leqslant \frac{\lg^2 n}{\ell^3} \ell^{L+\epsilon} \leqslant \frac{1}{\ell^{3/2}}.$$

Notice finally that

$$\frac{n}{\lg m} \sim \frac{n}{2b} \sim \frac{n}{\lg^2 n} \leqslant \frac{N}{5} \leqslant \frac{2n}{\lg^2 n} \sim \frac{n}{b} \sim \frac{2n}{\lg m}.$$

Step 1: Kronecker segmentation. For sufficiently large n, the last inequality implies that $n \leqslant (N/2)\, b$. In order to multiply two n-bit integers i and j, we expand them in base 2^b, i.e.

$$i = \sum_{k=0}^{N/2-1} i_k 2^{bk}, \qquad j = \sum_{k=0}^{N/2-1} j_k 2^{bk}, \qquad 0 \leqslant i_k, j_k < 2^b,$$

and form the polynomials

$$I = \sum_{k=0}^{N/2-1} i_k z^k, \qquad J = \sum_{k=0}^{N/2-1} j_k z^k.$$

Let $\bar{I}, \bar{J} \in (\mathbb{Z}/m\mathbb{Z})[z]_N^\ominus$ be the reductions of these polynomials modulo m and $z^N + 1$. Since $\deg (IJ) < N$ and $(IJ)_k < 2^{2b} N/2 \leqslant m$ for $k = 0, \ldots, N-1$, we can read off the product IJ from the product $\bar{I}\bar{J}$, after which $ij = (IJ)(2^b)$. Computing $I, J, \bar{I}, \bar{J}$ from i and j clearly takes linear time $O(n)$, and so does the retrieval of IJ and ij from $\bar{I}\bar{J}$. In other words, we have shown that

$$\mathsf{I}(n) \;\leqslant\; \mathsf{M}^\ominus_{\mathbb{Z}/m\mathbb{Z}}(N) + O(n). \tag{6.1}$$

Step 2: CRT transforms. At a second stage, we use negacyclic CRT transforms (section 3.5) followed by traditional CRT transforms (section 3.4) in order to reduce multiplication in $(\mathbb{Z}/m\mathbb{Z})[z]_N^\ominus$ to multiplication in $\mathbb{A}[\boldsymbol{x}]_{\boldsymbol{p}}^\circ$, where $\boldsymbol{x} = (x_1, \ldots, x_d)$, $\boldsymbol{p} = (p_1, \ldots, p_d)$, and $\mathbb{A} = (\mathbb{Z}/m\mathbb{Z})[u]_s^\ominus$:

$$\begin{aligned} \mathsf{M}^\ominus_{\mathbb{Z}/m\mathbb{Z}}(N) &\;\leqslant\; \mathsf{M}^\circ_{\mathbb{A}}(p_1 \cdots p_d) + O(N \lg N \lg m) \\ \mathsf{M}^\circ_{\mathbb{A}}(p_1 \cdots p_d) &\;\leqslant\; \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{p}) + O(|\boldsymbol{p}| \lg |\boldsymbol{p}| s \lg m). \end{aligned}$$

Since $O(|\boldsymbol{p}| \lg |\boldsymbol{p}| s \lg m) = O(N \lg |\boldsymbol{p}| \lg m) = O(N \lg N \lg m) = O(n \lg n)$, this yields

$$\mathsf{M}^\ominus_{\mathbb{Z}/m\mathbb{Z}}(N) \;\leqslant\; \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{p}) + O(n \lg n).$$

The computation of a generator of the cyclic group $\mathbb{F}_\pi^*$ can be done in time $O(\pi^{1/4+\epsilon}) = O(n^{(L^2+4\epsilon L)/4+\epsilon}) = O(n)$, by [53]. The lift of the $((\pi-1)/\nu)$-th power of this generator to $\mathbb{Z}/m\mathbb{Z}$ can be computed in time $(\log n)^{O(1)}$ and yields a primitive ν-th root of unity $\omega \in \mathbb{Z}/m\mathbb{Z}$. We perform the multivariate cyclic convolutions in $\mathbb{A}[\boldsymbol{x}]_{\boldsymbol{p}}^\circ$ using fast Fourier multiplication, where the discrete Fourier transforms are computed with respect to $\boldsymbol{\omega} = (\omega^{\nu/p_1}, \ldots, \omega^{\nu/p_d})$. In view of (3.8), this leads to

$$\begin{aligned} \mathsf{M}^\ominus_{\mathbb{Z}/m\mathbb{Z}}(N) &\;\leqslant\; 3\,\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}|\,(\mathsf{m}_{\mathbb{A}} + \mathsf{d}_{\mathbb{A},\mathbb{Z}}) + O(n \lg n) \\ &\;\leqslant\; 3\,\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}|\,\mathsf{m}_{\mathbb{A}} + O(n \lg n). \end{aligned}$$

Indeed, scalar divisions by integers in $\mathbb{A}$ can be performed in time

$$\mathsf{d}_{\mathbb{A},\mathbb{Z}} \;\leqslant\; s\,\mathsf{d}_{\mathbb{Z}/m\mathbb{Z},\mathbb{Z}} \;=\; O\left(\mathsf{s}_{\mathbb{A}} \frac{\mathsf{I}(\lg m)}{\lg m}\right) \;=\; O\left(\frac{n}{|\boldsymbol{p}|} (\lg \lg m)^{1+o(1)}\right) \;=\; O\left(\frac{n \lg n}{|\boldsymbol{p}|}\right)$$

using Schönhage–Strassen's algorithm for integer multiplication. Now we observe that $\lg s \asymp \lg n \prec \lg^2 n \asymp \lg m$ and $s \lg m = (N/|\boldsymbol{p}|) \lg m \leqslant (10 + o(1))\, n/|\boldsymbol{p}|$, whence $s\,\mathsf{I}(\lg m) = O\Big(\frac{n}{|\boldsymbol{p}|} \lg n\Big)$, again using Schönhage–Strassen multiplication. Performing multiplications in $\mathbb{A}$ using Kronecker multiplication, it also follows that

$$\mathsf{m}_{\mathbb{A}} \leqslant \mathsf{I}(s\,(2 \lg m + \lg s)) + O(s\,\mathsf{I}(\lg m)) \leqslant \mathsf{I}(3 s \lg m) + O\Big(\frac{n}{|\boldsymbol{p}|} \lg n\Big),$$

for sufficiently large n. We conclude that

$$\mathsf{M}^{\ominus}_{\mathbb{Z}/m\mathbb{Z}}(N) \;\leqslant\; 3\,\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}|\,\mathsf{I}(3 s \lg m) + O(n \lg n). \tag{6.2}$$

Step 3: multivariate Rader reduction. We compute the multivariate discrete Fourier transforms using Rader reduction. For each $\boldsymbol{w} \in \mathcal{W} = \{0,1\}^d$, let $\boldsymbol{r}_{\boldsymbol{w}} = (r_{\boldsymbol{w},1}, \ldots, r_{\boldsymbol{w},d})$ be such that $r_{\boldsymbol{w},k} = 1$ if $w_k = 0$ and $r_{\boldsymbol{w},k} = p_k - 1$ if $w_k = 1$ for $k = 1, \ldots, d$. Then (4.3) implies

$$\begin{aligned} \mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) \;&\leqslant\; \sum_{\boldsymbol{w} \in \mathcal{W}} \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) + O(|\boldsymbol{p}| \lg |\boldsymbol{p}|\, s \lg m) \\ &=\; \sum_{\boldsymbol{w} \in \mathcal{W}} \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) + O(n \lg n). \end{aligned}$$

The necessary precomputations for each of the $2^d = O(1)$ Rader reductions can be performed over $\mathbb{Z}/m\mathbb{Z}$ instead of $\mathbb{A}$. Using Schönhage–Strassen multiplication, this can be done in time

$$\begin{aligned} O(|\boldsymbol{p}|\,\mathsf{m}_{\mathbb{Z}/m\mathbb{Z}} + |\boldsymbol{p}| \lg |\boldsymbol{p}|\,\mathsf{s}_{\mathbb{Z}/m\mathbb{Z}}) \;&=\; O(|\boldsymbol{p}| \lg m\, (\lg \lg m)^{1+o(1)} + |\boldsymbol{p}| \lg |\boldsymbol{p}| \lg m) \\ &=\; O(|\boldsymbol{p}|\, s \lg |\boldsymbol{p}| \lg m) \\ &=\; O(n \lg n), \end{aligned}$$

since

$$(\lg \lg m)^{1+o(1)} \;=\; (\lg \lg n)^{1+o(1)} \;=\; O(\lg n) \;=\; O(\lg N) \;=\; O(\lg(s\,|\boldsymbol{p}|)) \;=\; O(s \lg |\boldsymbol{p}|).$$

Now for any $\boldsymbol{w} \in \mathcal{W} \setminus \{\boldsymbol{1}\}$, we have $|\boldsymbol{r}_w| \leqslant p_1 \cdots p_d / \ell$. Performing multiplications in $\mathbb{A}[\boldsymbol{u}]^{\circ}_{\boldsymbol{r}_w}$ using d-fold Kronecker substitution and Cantor–Kaltofen's variant of Schönhage–Strassen multiplication over $\mathbb{Z}/m\mathbb{Z}$, this yields

$$\begin{aligned} \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) \;&=\; O(\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}(2^d s\,|\boldsymbol{r}_{\boldsymbol{w}}|)) \\ &=\; O(2^d s\,|\boldsymbol{r}_{\boldsymbol{w}}| \lg m\, (\log (s\,|\boldsymbol{r}_{\boldsymbol{w}}| \lg m))^{1+o(1)}) \\ &=\; O(2^d\, (n/\ell)\, (\log n)^{1+o(1)}), \end{aligned}$$

whence

$$\sum_{\boldsymbol{w} \in \mathcal{W} \setminus \{\boldsymbol{1}\}} \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) \;=\; O(4^d\, (n/\ell)\, (\log n)^{1+o(1)}).$$

Since $4^d = O(1)$ and $1/\ell \leqslant 1/n^{\epsilon}$, the right hand side is bounded by $O(n)$, whence

$$\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) \;\leqslant\; \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{1}}) + O(n \lg n). \tag{6.3}$$

Step 4: second CRT transforms. For $k = 1, \ldots, d$, we may decompose $r_{\boldsymbol{1},k} = p_k - 1 = \ell\, q_k$, where q_k is odd. Using multivariate CRT transforms (Corollary A.8), this yields

$$\mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{1}}) \;\leqslant\; \mathsf{M}^{\circ}_{\mathbb{A}}(q_1, \ldots, q_d, \ell, \ldots, \ell) + O(n \lg n).$$

Setting $\mathbb{B} = \mathbb{A}[\boldsymbol{y}]_{\boldsymbol{q}}^{\circ}$ and $\boldsymbol{\ell} = \big(\ell, \overset{d\times}{\ldots}, \ell\big)$, this relation can be rewritten as

$$\mathsf{M}_{\mathbb{A}}^{\circ}(\boldsymbol{r_1}) \leqslant \mathsf{M}_{\mathbb{B}}^{\circ}(\boldsymbol{\ell}) + O(n \lg n). \tag{6.4}$$

Step 5: Nussbaumer polynomial transforms. Since $\ell \mid s$, we have a fast root of unity $u^{2s/\ell}$ of order ℓ in $\mathbb{A}$ and in $\mathbb{B}$. We may thus compute polycyclic convolutions of order $\boldsymbol{\ell}$ over $\mathbb{B}$ using fast Fourier multiplication, with Nussbaumer polynomial transforms in the role of discrete Fourier transforms. From (3.10), it follows that

$$\begin{aligned}\mathsf{M}_{\mathbb{B}}^{\circ}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\,(\mathsf{m}_{\mathbb{B}} + \mathsf{d}_{\mathbb{B},\mathbb{Z}}) + O(|\boldsymbol{\ell}| \lg |\boldsymbol{\ell}|\, \mathsf{s}_{\mathbb{B}}) \\ &\leqslant |\boldsymbol{\ell}|\,\mathsf{m}_{\mathbb{B}} + O(|\boldsymbol{\ell}| \lg |\boldsymbol{\ell}|\, \mathsf{s}_{\mathbb{B}}) \\ &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}_{\mathbb{A}}^{\circ}(\boldsymbol{q}) + O(n \lg n).\end{aligned}$$

Here we used the bound $\mathsf{d}_{\mathbb{B},\mathbb{Z}} = O(\mathsf{s}_{\mathbb{B}} \lg |\boldsymbol{\ell}|)$, which is shown in a similar way as above. Modulo a change of variables $u \to \eta u$ where $\eta = \omega^{\nu/(2s)}$ is a principal $(2s)$-th root of unity in $\mathbb{Z}/m\mathbb{Z}$, negacyclic convolutions in $(\mathbb{Z}/m\mathbb{Z})[u]_s^{\ominus}$ reduce to cyclic convolutions in $(\mathbb{Z}/m\mathbb{Z})[u]_s^{\circ}$. Combining this with the above relations, this yields

$$\begin{aligned}\mathsf{M}_{\mathbb{B}}^{\circ}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}^{\circ}(s, \boldsymbol{q}) + O(|\boldsymbol{\ell}|\, s\, |\boldsymbol{q}|\, \mathsf{m}_{\mathbb{Z}/m\mathbb{Z}}) + O(n \lg n) \\ &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}^{\circ}(s, \boldsymbol{q}) + O(n \lg n).\end{aligned}$$

The bound $|\boldsymbol{\ell}|\, s\, |\boldsymbol{q}|\, \mathsf{m}_{\mathbb{Z}/m\mathbb{Z}} \leqslant N\, \mathsf{m}_{\mathbb{Z}/m\mathbb{Z}} \leqslant O(n\, \mathsf{m}_{\mathbb{Z}/m\mathbb{Z}} / \lg m) \leqslant O(n \lg n)$ follows from (1.1).

Step 6: residual cyclic convolutions. By construction, $\mathbb{Z}/m\mathbb{Z}$ contains principal roots of unity of orders $s, q_1, \ldots, q_d$, which can again be computed in time $O(n)$ using a similar method as above. We may thus compute cyclic convolutions of order $(s, \boldsymbol{q})$ over $\mathbb{Z}/m\mathbb{Z}$ using fast Fourier multiplication. Since $|\boldsymbol{\ell}|\, s\, |\boldsymbol{q}|\, \mathsf{m}_{\mathbb{Z}/m\mathbb{Z}} = O(n \lg n)$ and $|\boldsymbol{\ell}|\, s\, |\boldsymbol{q}|\, \mathsf{d}_{\mathbb{Z}/m\mathbb{Z},\mathbb{Z}} = O(n \lg n)$, we obtain

$$\mathsf{M}_{\mathbb{B}}^{\circ}(\boldsymbol{\ell}) \leqslant 3N \left(\frac{\mathsf{F}_{\mathbb{Z}/m\mathbb{Z}}(s)}{s} + \sum_{k=1}^{d} \frac{\mathsf{F}_{\mathbb{Z}/m\mathbb{Z}}(q_k)}{q_k} \right) + O(n \lg n).$$

Using Bluestein reduction (2.10), the univariate discrete Fourier transforms can be transformed back into cyclic products:

$$\mathsf{M}_{\mathbb{B}}^{\circ}(\boldsymbol{\ell}) \leqslant 3N \left(\frac{\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}^{\circ}(s)}{s} + \sum_{k=1}^{d} \frac{\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}^{\circ}(q_k)}{q_k} \right) + O(n \lg n).$$

We perform the cyclic products using Kronecker substitution. Since I is increasing (by definition), this yields

$$\begin{aligned}\mathsf{M}_{\mathbb{B}}^{\circ}(\boldsymbol{\ell}) &\leqslant 3N \left(\frac{\mathsf{I}(s\,(2 \lg m + \lg s))}{s} + \sum_{k=1}^{d} \frac{\mathsf{I}(q_k\,(2 \lg m + \lg q_k))}{q_k} \right) + O(n \lg n). \\ &\leqslant 9N \lg m \left(\frac{\mathsf{I}(3s \lg m)}{3s \lg m} + \sum_{k=1}^{d} \frac{\mathsf{I}(3q_k \lg m)}{3q_k \lg m} \right) + O(n \lg n). \end{aligned} \tag{6.5}$$

Step 7: recurse. Combining the relations (6.1), (6.2), (6.3), (6.4) and (6.5), while using the fact that $27N \lg m \leqslant (270 + o(1))\, n \leqslant 271\, n$ for sufficiently large n, we obtain

$$\mathsf{I}(n) \leqslant 271\, n \left(\frac{\mathsf{I}(3s \lg m)}{3s \lg m} + \sum_{k=1}^{d} \frac{\mathsf{I}(3q_k \lg m)}{3q_k \lg m} \right) + |\boldsymbol{p}|\, \mathsf{I}(3s \lg m) + O(n \lg n).$$

Using that $3|\boldsymbol{p}|s \lg m \leqslant (30 + o(1))\, n \leqslant 31\, n$, this relation simplifies to

$$\mathsf{I}(n) \;\leqslant\; 302\, n \left(\frac{\mathsf{I}(3s \lg m)}{3s \lg m} + \sum_{k=1}^{d} \frac{\mathsf{I}(3q_k \lg m)}{3q_k \lg m} \right) + O(n \lg n).$$

In terms of the normalized cost function

$$\mathsf{I}^*(n) \;:=\; \max_{2 \leqslant k \leqslant n} \frac{\mathsf{I}(k)}{k \log k}$$

and $M = 3 \max(s, q_1, \ldots, q_k) \lg m$, this relation becomes

$$\begin{aligned} \frac{\mathsf{I}(n)}{n \log n} &\leqslant \frac{302\, \mathsf{I}^*(M)}{\log n} \left(\log(3s \lg m) + \sum_{k=1}^{d} \log(3q_k \lg m) \right) + O(1) \\ &\leqslant \frac{302\, \mathsf{I}^*(M)}{\log n} (\log(s q_1 \cdots q_d) + (d+1) \log(3 \lg m)) + O(1). \end{aligned}$$

By Lemma 5.1, we have $q_k = O(\ell^{L-1} (\log \ell)^L)$ for all k. For sufficiently large n, this means that $q_k \leqslant {}^1\!/_8\, (\ell/2)^{L+\epsilon-1}$, whence $s q_1 \cdots q_d \leqslant (\ell/2)^{3+(L+\epsilon-1)d} \leqslant n^{L-1+4\epsilon}$ and $\log(s q_1 \cdots q_d) \leqslant (L-1+4\epsilon) \log n$. Using that $(d+1) \log(3 \lg m) = O(\log \log n)$, the latest bound on $\mathsf{I}(n)/(n \log n)$ therefore further simplifies into

$$\frac{\mathsf{I}(n)}{n \log n} \;\leqslant\; \mathsf{I}^*(M)\, (302\, (L-1+4\epsilon) + o(1)) + B,$$

for some suitable universal constant $B > 0$. For sufficiently large $n \geqslant n_0$, using our assumptions $L < 1 + {}^1\!/_{303}$, we have $n > M$ and $302\, (L-1+4\epsilon) + o(1) < 1 - \epsilon$, whence

$$\frac{\mathsf{I}(n)}{n \log n} \;\leqslant\; (1-\epsilon)\, \mathsf{I}^*(M) + B.$$

Let us show by induction over n that this implies $\mathsf{I}^*(n) \leqslant \mathsf{I}^*(n_0) + B/\epsilon$ for all $n \geqslant n_0$. This is clear for $n = n_0$, so assume that $n > n_0$. Then the above inequality yields

$$\frac{\mathsf{I}(n)}{n \log n} \;\leqslant\; (1-\epsilon) \left(\mathsf{I}^*(n_0) + \frac{B}{\epsilon} \right) + B \;=\; (1-\epsilon)\, \mathsf{I}^*(n_0) + \frac{B}{\epsilon} \;\leqslant\; \mathsf{I}^*(n_0) + \frac{B}{\epsilon}.$$

Consequently, $\mathsf{I}^*(n) = \max(\mathsf{I}^*(n-1), \mathsf{I}(n)/(n \log n)) \leqslant \mathsf{I}^*(n_0) + B/\epsilon$. This completes the induction and we conclude that $\mathsf{I}^*(n) = O(1)$, whence $\mathsf{I}(n) = O(n \log n)$.

Remark 6.1. In our proof, we have not attempted to optimize the constant 303. With a bit more work, it should be possible to achieve any constant larger than 32, as follows:

- On three occasions, we used the crude estimate $o(1) \leqslant 1$ for large n; this explains the increase from 300 to 303.
- Taking $\epsilon \asymp 1/\log \log \log n$, the cost of the pointwise multiplications in (6.2) becomes negligible, so there is no further increase from 271 to 302 in the last step. Since this also means that the dimension d is no longer constant, a few adaptations are necessary. For instance, we need Remark 5.2 that $P_\nu(k) = O((k\, (\nu + \log k))^L)$ and at the end of step 3, we only have $4^d = O(\log \log n)$.
- Taking $s := 2^{\lg((4+\epsilon)n/(p_1 \cdots p_d \lg^2 n))}$ one gains a factor $5/4$.
- In recursive FFT-multiplications, one argument is always fixed (due to the fact that one of the multipliers in the Bluestein and Rader reductions is fixed). In a similar way as in [25], this makes it possible to save one fast Fourier transform out of three. Since this observation can be used twice in steps 2 and 6, a further factor of $9/4$ can be gained in this way.

- In step 6, we used the crude bound $2s(\lg m + \lg s) \leqslant 3s \lg m$. Replacing the right-hand side by $(2+\epsilon)s\lg m$, we save another factor $3/2$.
- By allowing s to contain some small odd factors, we may further improve the inequality $N \lg m \leqslant (10+o(1))n$ in step 7 to become $N \lg m \leqslant (5+o(1))n$. This saves another factor of two.

We finally recall that we opted to present our algorithm for number theoretic FFTs. When switching to approximate complex FFTs, one advantage is that one may work more extensively with primitive roots of power of two orders. This should also allow for the use of Crandall–Fagin's reduction (in a similar way as in [25]) to further release the assumption on the Linnik constant to $L < 1 + {}^1\!/_{16}$.

7. PROOF OF THEOREM 1.2 IN CHARACTERISTIC $\pi \neq 2$

Recall that elements of the finite field $\mathbb{F}_{\pi^\kappa}$ are represented as remainder classes of polynomials in $\mathbb{F}_\pi[x]_\kappa$ modulo μ, for some given monic irreducible polynomial $\mu \in \mathbb{F}_\pi[x]$ of degree κ. Using Kronecker substitution and segmentation, one has the following well known complexity bounds [27, section 3]:

$$\begin{aligned} \mathsf{M}_{\pi^\kappa}(n) &\leqslant \mathsf{M}_\pi(2n\kappa) + O(n\mathsf{M}_\pi(\kappa)) \\ \mathsf{M}_\pi(n) &\leqslant \mathsf{M}_{\pi^\kappa}(\lceil n/\lfloor \kappa/2 \rfloor \rceil) + O(n \lg \pi). \end{aligned}$$

In order to establish Theorem 1.2, it therefore suffices to consider the case when $q = \pi$ is a prime number. In this section, we first prove the theorem in the case when $\pi \neq 2$. The case $\pi = 2$ is treated in the next section. If $\lg n = O(\lg \pi)$, then it is well known that $\mathsf{M}_\pi(n) = O(\mathsf{I}(n \lg \pi))$, again using Kronecker substitution. In this particular case, Theorem 1.2 therefore follows from Theorem 1.1. Let $L > 1$ be a Linnik constant with $L < 1 + 2^{-777}$. Our algorithm proceeds as follows.

Step 0: setting up the parameters. Assume that we wish to multiply two polynomials of degree $< n$ in $\mathbb{F}_\pi[x]$. For $n \geqslant n_0 := \tau\pi^2 \geqslant \pi$ with $\tau \geqslant 1$ and suitable parameters κ and N, we will reduce this problem to a multiplication problem in $\mathbb{F}_{\pi^\kappa}[z]_N^\ominus$. If $n < n_0$, then we use Kronecker multiplication. The factor τ is a sufficiently large absolute constant such that various asymptotic inequalities encountered during the proof hold for all $n \geqslant n_0$. The other parameters are determined as follows:

- $\epsilon := 1/770$.
- $\ell := 2^{\lg(n^\epsilon)}$ is the smallest power of two with $\ell \geqslant n^\epsilon > 1$.
- With the notations from section 5.3 and for each $\nu > 0$, let $p_\nu = q_\nu \ell + 1$ be the ν-th element in the list $P_1^*(1,\ell), P_2^*(1,\ell), P_3^*(1,\ell), \ldots$ with $p_\nu \neq \pi$ and $2 \nmid q_\nu$.
- $d := \min\{d \in \mathbb{N} : \ell^4 p_1 \cdots p_d \geqslant n\}$.
- $\kappa := \operatorname{lcm}(p_1 - 1, \ldots, p_d - 1)$.
- $s := 2^{\lg(4n/(\kappa p_1 \cdots p_d))}$.
- $N := s p_1 \cdots p_d$.

Since $\ell^4 p_1 \cdots p_{766} \geqslant \ell^{770} = \ell^{1/\epsilon} \geqslant n$, we must have $d \leqslant 766$. From Lemma 5.4, we get

$$\ell < p_1 < \cdots < p_d \leqslant P_{d+2}^*(1,\ell) < \bar{C}\,\ell^{\bar{L}} \leqslant \ell^{1+\epsilon},$$

for sufficiently large n, where

$$\bar{L} := 1 + (L+1)^{d+1}(L-1) < 1 + (2+2^{-777})^{767} 2^{-777} < 1 + \epsilon,$$

and $\bar{C}$ is a constant. As in the previous section, the prime numbers $p_1, \ldots, p_d$ and the rest of the parameters can be computed in linear time $O(n)$. Let us proceed with a few useful bounds that hold for sufficiently large n. From the definitions of s and N, we clearly have

$$4n \leqslant \kappa N \leqslant 8n.$$

From $p_d < \ell^{1+\epsilon}$ and $\kappa = \ell \operatorname{lcm}(q_1, \ldots, q_d) \leqslant \ell^{1-d} p_1 \cdots p_d$, we get

$$\ell \leqslant \kappa \leqslant \ell^{1+d\epsilon} \leqslant \ell^2$$

and

$$\ell \leqslant 4\,\ell^{2-(d+1)\epsilon} = 4\,\ell^{4-(1+\epsilon)-(1+d\epsilon)} \leqslant \frac{4\,\ell^{4-(1+\epsilon)}}{\kappa} \leqslant \frac{4n}{\kappa p_1 \cdots p_d} \leqslant s \leqslant \frac{8n}{\kappa p_1 \cdots p_d} \leqslant \frac{8\,\ell^4}{\kappa} \leqslant 8\,\ell^3.$$

Step 0, continued: setting up the FFT field. Notice that $\ell \mid \kappa$. Elements of the field $\mathbb{F}_{\pi^\kappa}$ will be represented as polynomials in $\mathbb{F}_\pi[x]$ modulo μ, where $\mu \in \mathbb{F}_\pi[x]$ is a monic irreducible polynomial of degree κ. By [51, Theorem 3.2], such a polynomial can be computed in time $O(\pi^{1/2}(\lg \pi)^3 \kappa^{3+\epsilon} + (\lg \pi)^2 \kappa^{4+\epsilon}) = O(\pi^{1/2+\epsilon} \kappa^{4+\epsilon}) = O(n^{1/2+\epsilon} \ell^{8+2\epsilon}) = O(n^{1/2+10\epsilon}) = O(n)$, where we used the assumption that $n \geqslant \pi$.

Setting $\nu := p_1 \cdots p_d$, we claim that the field $\mathbb{F}_{\pi^\kappa}$ admits a primitive ν-th root of unity ω, that can also be computed in time $O(n)$. Indeed, $p_i - 1$ divides κ for each $i \in \{1, \ldots, d\}$ and $p_i \neq \pi$, whence $\pi^\kappa \equiv 1 \bmod p_i$, so that $p_i \mid (\pi^\kappa - 1)$. We compute ω by brute force: for all polynomials $P \in \mathbb{F}_\pi[x]$, starting with those of smallest degree, we check whether $P \bmod \mu$ is a primitive ν-th root. By [52, Theorem 1.1, $r \leqslant \kappa \lg \pi$], this procedure succeeds after at most $O(\pi\,(\kappa \lg \pi)^{4+\epsilon}\,\kappa^2)$ checks. In order to check whether a given $\omega = P \bmod \mu$ is a primitive ν-th root of unity, it suffices to verify that $\omega^{\nu/p_i} \neq 1$ for each $i \in \{1, \ldots, d\}$, which can be done in time $O((\kappa \lg \pi)^{1+\epsilon} \lg n)$ using binary powering. The total time to compute a suitable ω is therefore bounded by $O(\pi\,(\kappa \lg \pi)^{5+2\epsilon} \kappa^2 \lg n) = O(\pi n^{2(7+2\epsilon)\epsilon} (\lg n)^{6+2\epsilon}) = O(n)$, using our assumption that $n \geqslant \pi^2$.

Step 1: Kronecker segmentation. Recall that $\kappa N \geqslant 4n$. In order to multiply two polynomials $a, b \in \mathbb{F}_\pi[x]$ of degree $< n$, we cut them in chunks $a_k, b_k \in \mathbb{F}_\pi[x]$ of degree $< \kappa/2$, i.e.

$$a = \sum_{k=0}^{N/2-1} a_k x^{(\kappa/2)k}, \qquad b = \sum_{k=0}^{N/2-1} b_k x^{(\kappa/2)k},$$

and form the polynomials

$$A = \sum_{k=0}^{N/2-1} a_k z^k, \qquad B = \sum_{k=0}^{N/2-1} b_k z^k.$$

Let $\bar{A}, \bar{B} \in \mathbb{F}_{\pi^\kappa}[z]_N^\ominus$ be the reductions of these polynomials modulo μ and $z^N + 1$. Since $\deg_z(AB) < N$ and $\deg_x(AB) < \kappa$, we can read off the product AB from the product $\bar{A}\bar{B}$, after which $ab = (AB)(x^{\kappa/2})$. Computing $A, B, \bar{A}, \bar{B}$ from a and b clearly takes linear time $O(n \lg \pi)$, and so does the retrieval of AB and ab from $\bar{A}\bar{B}$. In other words, we have shown that

$$\mathsf{M}_\pi(n) \;\leqslant\; \mathsf{M}^\ominus_{\pi^\kappa}(N) + O(n \lg \pi). \tag{7.1}$$

Step 2: CRT transforms. At a second stage, we use negacyclic CRT transforms followed by traditional CRT transforms in order to reduce multiplication in $\mathbb{F}_{\pi^\kappa}[z]_N^\ominus$ to multiplication in $\mathbb{A}[\boldsymbol{x}]_{\boldsymbol{p}}^\circ$, where $\boldsymbol{x} = (x_1, \ldots, x_d)$, $\boldsymbol{p} = (p_1, \ldots, p_d)$, and $\mathbb{A} = \mathbb{F}_{\pi^\kappa}[u]_s^\ominus$:

$$\begin{aligned} \mathsf{M}^\ominus_{\pi^\kappa}(N) &\leqslant \mathsf{M}^\circ_{\mathbb{A}}(p_1 \cdots p_d) + O(N \lg N \kappa \lg \pi) \\ \mathsf{M}^\circ_{\mathbb{A}}(p_1 \cdots p_d) &\leqslant \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{p}) + O(|\boldsymbol{p}| \lg |\boldsymbol{p}| s \kappa \lg \pi). \end{aligned}$$

Since $O(|\boldsymbol{p}| \lg |\boldsymbol{p}| s \kappa \lg \pi) = O(N \lg |\boldsymbol{p}| \kappa \lg \pi) = O(N \lg N \kappa \lg \pi) = O(n \lg \pi \lg n)$, this yields

$$\mathsf{M}^\ominus_{\pi^\kappa}(N) \leqslant \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{p}) + O(n \lg \pi \lg n).$$

We perform the multivariate cyclic convolutions in $\mathbb{A}[\boldsymbol{x}]_{\boldsymbol{p}}^\circ$ using fast Fourier multiplication, where the discrete Fourier transforms are computed with respect to $\boldsymbol{\omega} = (\omega^{\nu/p_1}, \ldots, \omega^{\nu/p_d})$. In view of (3.8) and

$$|\boldsymbol{p}| \mathsf{d}_{\mathbb{A},\mathbb{Z}} \leqslant |\boldsymbol{p}| \mathsf{s}_{\mathbb{A}} \frac{\mathsf{d}_{\mathbb{Z}/\pi\mathbb{Z},\mathbb{Z}}}{\lg \pi} = O(n \lg \pi \, (\lg \lg \pi)^{1+o(1)}) = O(n \lg \pi \lg n),$$

this yields

$$\begin{aligned} \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{p}) &\leqslant 3\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}| (\mathsf{m}_{\mathbb{A}} + \mathsf{d}_{\mathbb{A},\mathbb{Z}}) + O(|\boldsymbol{p}| \mathsf{s}_{\mathbb{A}}) \\ &\leqslant 3\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}| \mathsf{m}_{\mathbb{A}} + O(n \lg \pi \lg n). \end{aligned}$$

The reduction of a polynomial of degree $<2\kappa$ in $\mathbb{F}_\pi[x]$ modulo μ using Barrett's algorithm [3] essentially requires two polynomial multiplications in $\mathbb{F}_\pi[x]$ of degree $<\kappa$, modulo the precomputation of a preinverse in time $O(\mathsf{M}_\pi(\kappa)) = O(n)$. Performing multiplications in $\mathbb{A}$ using Kronecker substitution, we thus have

$$\mathsf{m}_{\mathbb{A}} \leqslant \mathsf{M}_\pi(2\kappa s) + 2s \mathsf{M}_\pi(\kappa) + O(\kappa s \lg \pi).$$

For sufficiently large n, we conclude that

$$\mathsf{M}^\ominus_{\pi^\kappa}(N) \leqslant 3\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}| \mathsf{M}_\pi(2\kappa s) + 2|\boldsymbol{p}| s \mathsf{M}_\pi(\kappa) + O(n \lg \pi \lg n). \tag{7.2}$$

Step 3: multivariate Rader reduction. We compute the multivariate discrete Fourier transforms using Rader reduction. For each $\boldsymbol{w} \in \mathcal{W} = \{0,1\}^d$, let $\boldsymbol{r}_{\boldsymbol{w}} = (r_{\boldsymbol{w},1}, \ldots, r_{\boldsymbol{w},d})$ be such that $r_{\boldsymbol{w},k} = 1$ if $w_k = 0$ and $r_{\boldsymbol{w},k} = p_k - 1$ if $w_k = 1$ for $k = 1, \ldots, d$. Then

$$\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) = \sum_{\boldsymbol{w} \in \mathcal{W}} \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) + O(n \lg \pi \lg n).$$

The necessary precomputations for each of the $2^d = O(1)$ Rader reductions can be performed over $\mathbb{F}_{\pi^\kappa}$ instead of $\mathbb{A}$, in time

$$\begin{aligned} O(|\boldsymbol{p}| \mathsf{m}_{\mathbb{F}_{\pi^\kappa}} + |\boldsymbol{p}| \lg |\boldsymbol{p}| \mathsf{s}_{\mathbb{F}_{\pi^\kappa}}) &= O(|\boldsymbol{p}| \lg \pi^\kappa (\lg \lg \pi^\kappa)^{1+o(1)} + |\boldsymbol{p}| \lg |\boldsymbol{p}| \lg \pi^\kappa) \\ &= O(\kappa |\boldsymbol{p}| \lg \pi \, (\lg n)^{1+o(1)} + \kappa |\boldsymbol{p}| \lg \pi \lg n) \\ &= O(\kappa s |\boldsymbol{p}| \lg \pi \lg n) \\ &= O(n \lg \pi \lg n). \end{aligned}$$

Now for any $\boldsymbol{w} \in \mathcal{W} \setminus \{\boldsymbol{1}\}$, we have $|\boldsymbol{r}_w| \leqslant p_1 \cdots p_d / \ell$. Performing multiplications in $\mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}})$ using d-fold Kronecker substitution and (1.6), this yields

$$\begin{aligned} \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) &= O(\mathsf{M}_\pi(2^d \kappa s |\boldsymbol{r}_{\boldsymbol{w}}|)) \\ &= O(2^d \kappa s |\boldsymbol{r}_{\boldsymbol{w}}| \lg \pi \, (\log (2^d \kappa s |\boldsymbol{r}_{\boldsymbol{w}}| \lg \pi))^{1+o(1)}) \\ &= O(2^d (n \lg \pi / \ell) \, (\log (n \lg \pi))^{1+o(1)}), \end{aligned}$$

whence

$$\sum_{\boldsymbol{w} \in \mathcal{U} \setminus \{\mathbf{1}\}} \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\boldsymbol{w}}) \;=\; O(4^d\,(n \lg \pi / \ell)\,(\log\,(n \lg \pi))^{1+o(1)}).$$

Since $4^d = O(1)$, $1/\ell \leqslant 1/n^{\epsilon}$, and $n \geqslant \lg \pi$, the right hand side is bounded by $O(n \lg \pi)$, whence

$$\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) \;\leqslant\; \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\mathbf{1}}) + O(n \lg \pi \lg n). \tag{7.3}$$

Step 4: second CRT transforms. For $k = 1, \ldots, d$, we may decompose $r_{\mathbf{1},k} = p_k - 1 = \ell\, q_k$, where q_k is odd. Using multivariate CRT transforms, this yields

$$\mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\mathbf{1}}) \;\leqslant\; \mathsf{M}^{\circ}_{\mathbb{A}}(q_1, \ldots, q_d, \ell, \ldots, \ell) + O(n \lg \pi \lg n).$$

Setting $\mathbb{B} = \mathbb{A}[\boldsymbol{y}]^{\circ}_{\boldsymbol{q}}$ and $\boldsymbol{\ell} = \big(\ell, \overset{d\times}{\ldots}, \ell\big)$, this relation can be rewritten as

$$\mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r}_{\mathbf{1}}) \;\leqslant\; \mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) + O(n \lg \pi \lg n). \tag{7.4}$$

Step 5: Nussbaumer polynomial transforms. Since $\ell \,|\, s$, we have a fast root of unity $u^{2s/\ell}$ of order ℓ in $\mathbb{A}$ and in $\mathbb{B}$. We may thus compute polycyclic convolutions of order $\boldsymbol{\ell}$ over $\mathbb{B}$ using fast Fourier multiplication, with Nussbaumer polynomial transforms in the role of discrete Fourier transforms:

$$\begin{aligned}
\mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\,(\mathsf{m}_{\mathbb{B}} + \mathsf{d}_{\mathbb{B},\mathbb{Z}}) + O(|\boldsymbol{\ell}| \lg |\boldsymbol{\ell}|\, \mathsf{s}_{\mathbb{B}}) \\
&\leqslant |\boldsymbol{\ell}|\, \mathsf{m}_{\mathbb{B}} + O(|\boldsymbol{\ell}| \lg |\boldsymbol{\ell}|\, \mathsf{s}_{\mathbb{B}}) \\
&\leqslant |\boldsymbol{\ell}|\, \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{q}) + O(n \lg \pi \lg n).
\end{aligned}$$

Step 6: residual cyclic convolutions. This time, we convert the residual multivariate cyclic convolutions of length $\boldsymbol{q}$ back into univariate cyclic convolutions of length $|\boldsymbol{q}|$ using CRT transforms the other way around. This is possible since $s, q_1, \ldots, q_d$ are pairwise coprime, and we get

$$\begin{aligned}
\mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{q}) &\leqslant \mathsf{M}^{\circ}_{\mathbb{A}}(|\boldsymbol{q}|) + O(\kappa s |\boldsymbol{q}| \lg \pi \lg |\boldsymbol{q}|) \\
&\leqslant \mathsf{M}^{\ominus}_{\pi^{\kappa}}(s|\boldsymbol{q}|) + O(\kappa s |\boldsymbol{q}| \lg \pi \lg (s|\boldsymbol{q}|)),
\end{aligned}$$

whence

$$\begin{aligned}
\mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\, \mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{q}) + O(n \lg \pi \lg n) \\
&\leqslant |\boldsymbol{\ell}|\, \mathsf{M}^{\ominus}_{\pi^{\kappa}}(s|\boldsymbol{q}|) + O(n \lg \pi \lg n).
\end{aligned}$$

We perform the univariate cyclic products using Kronecker substitution. Since reductions of polynomials in $\mathbb{F}_{\pi}[x]_{2\kappa}$ modulo μ can be done in time $2\,\mathsf{M}_{\pi}(\kappa) + O(\kappa \lg \pi)$, this yields

$$\begin{aligned}
\mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\, \mathsf{M}_{\pi}(2\kappa s |\boldsymbol{q}|) + 2|\boldsymbol{\ell}||\boldsymbol{q}|\, s\, \mathsf{M}_{\pi}(\kappa) + O(n \lg \pi \lg n) \\
&\leqslant 2N\kappa \frac{\mathsf{M}_{\pi}(2\kappa s |\boldsymbol{q}|)}{2\kappa s |\boldsymbol{q}|} + 2|\boldsymbol{p}|\, s\, \mathsf{M}_{\pi}(\kappa) + O(n \lg \pi \lg n).
\end{aligned} \tag{7.5}$$

Step 7: recurse. Combining the relations (7.1), (7.2), (7.3), (7.4) and (7.5), while using the fact that $6N\kappa \leqslant 48n$, we obtain

$$\mathsf{M}_{\pi}(n) \;\leqslant\; 48n \left(\frac{\mathsf{M}_{\pi}(2\kappa s |\boldsymbol{q}|)}{2\kappa s |\boldsymbol{q}|} \right) + |\boldsymbol{p}|\, \mathsf{M}_{\pi}(2\kappa s) + 8|\boldsymbol{p}|\, s\, \mathsf{M}_{\pi}(\kappa) + O(n \lg \pi \lg n).$$

In terms of the normalized cost function

$$\mathsf{M}^{*}_{\pi}(n) \;=\; \max_{2 \leqslant k \leqslant n} \frac{\mathsf{M}_{\pi}(k)}{k \lg \pi \log (k \lg \pi)}$$

and $M = 2\kappa s|\boldsymbol{q}|$, this relation becomes

$$\begin{aligned}\frac{\mathsf{M}_\pi(n)}{n\lg\pi\log(n\lg\pi)} &\leqslant 48\,\mathsf{M}^*_\pi(2\kappa s|\boldsymbol{q}|)\,\frac{\log(2\kappa s|\boldsymbol{q}|\lg\pi)}{\log(n\lg\pi)}\\ &\quad + 16\,\mathsf{M}^*_\pi(2\kappa s)\,\frac{\log(2\kappa s\lg\pi)}{\log(n\lg\pi)} + 64\,\mathsf{M}^*_\pi(\kappa)\,\frac{\log(\kappa\lg\pi)}{\log(n\lg\pi)} + O(1)\\ &\leqslant 128\,\mathsf{M}^*_\pi(M)\,\frac{\log(2\kappa s|\boldsymbol{q}|\lg\pi)}{\log(n\lg\pi)} + O(1),\end{aligned}$$

where we used the fact that $2|\boldsymbol{p}|\kappa s \leqslant 16n$. Recall that $s \leqslant 8\ell^3$ and $q_k < \ell^\epsilon$ for $k = 1,\ldots,d$, whence $s|\boldsymbol{q}| < 8\ell^{3+d\epsilon} \leqslant 8\ell^{4-4\epsilon}$, whereas $\kappa \leqslant \ell^2$. Consequently, for large n, we have

$$\begin{aligned}\log(2\kappa s|\boldsymbol{q}|\lg\pi) &\leqslant (6-4\epsilon)\,\epsilon\log n + O(\log\lg\pi)\\ &\leqslant (6-4\epsilon)\,\epsilon\log n + O(\log\lg n)\\ &\leqslant 6\epsilon\log n\\ &\leqslant 6\epsilon\log(n\lg\pi).\end{aligned}$$

For some suitable universal constant $B > 0$, this yields

$$\frac{\mathsf{M}_\pi(n)}{n\lg\pi\log(n\lg\pi)} \leqslant 768\,\mathsf{M}^*_\pi(M)\,\epsilon + B.$$

For sufficiently large n, we have $M = 2\kappa s|\boldsymbol{q}| \leqslant 16\frac{|\boldsymbol{q}|}{|\boldsymbol{p}|}n \leqslant 16n\ell^{-d} < n$ and $768\epsilon < 1-\epsilon$, whence

$$\frac{\mathsf{M}_\pi(n)}{n\lg\pi\log(n\lg\pi)} \leqslant (1-\epsilon)\,\mathsf{M}^*_\pi(M) + B.$$

In a similar way as in the previous section, this implies $\mathsf{M}^*_\pi(n) \leqslant \mathsf{M}^*_\pi(n_0) + B/\epsilon$ for all $n \geqslant n_0$. Now recall that Kronecker substitution and Theorem 1.1 imply $\mathsf{M}_\pi(n) = O(\mathsf{I}(n\lg\pi)) = O(n\lg\pi\log(n\lg\pi))$ for all $n \leqslant n_0 = O(\pi^2)$, whence $\mathsf{M}^*_\pi(n_0) = O(1)$. We conclude that $\mathsf{M}_\pi(n) = O(n\lg\pi\lg(n\lg\pi))$.

Remark 7.1. The hypothesis $L < 1 + 2^{-777}$ is very pessimistic and mainly due to the way we reduce the residual multivariate cyclic convolutions back to univariate ones. Our assumption that the $q_1,\ldots,q_d$ be pairwise coprime clearly makes this back-reduction very easy. Nevertheless, with more work, we expect that this assumption can actually be released. We intend to come back to this issue in a separate paper on the computation of multivariate cyclic convolution products.

8. PROOF OF THEOREM 1.2 IN CHARACTERISTIC $\pi = 2$

The proof of Theorem 1.2 in the case when $\pi = 2$ is similar to the one from the previous section, with two important changes:

- Following Schönhage [48], we need to use triadic DFTs instead of dyadic ones. In particular, $\ell := 3^{\lceil \log_3(n^\epsilon)\rceil}$ will now be a power of three.
- The transform lengths $p_i - 1$ are unavoidably even. This time, we will therefore have $p_i = 2q_i\ell + 1$, where the q_i are pairwise coprime. As a consequence, we will need a separate mechanism to deal with residual polycyclic convolution products of length 2 in each variable.

Given a ring R and an integer $t \geqslant 0$, it will be convenient to define

$$R_t := R[\theta_1,\ldots,\theta_t]/(\theta_1^2-1,\ldots,\theta_t^2-1).$$

We will also use the abbreviations $\mathbb{F}_{\pi^\kappa,t} := (\mathbb{F}_{\pi^\kappa})_t$, $\mathsf{M}_{R,t} := \mathsf{M}_{R_t}$, $\mathsf{M}_{\pi^\kappa,t} := \mathsf{M}_{\mathbb{F}_{\pi^\kappa},t}$, and so on. Throughout this section, we assume that $L > 1$ is a Linnik constant with $L < 1 + 2^{-1162}$. Our algorithm proceeds as follows.

Step 0: setting up the parameters. Let $t \geqslant 0$ and assume that we wish to multiply two polynomials of degree $< n$ in $\mathbb{F}_{\pi,t}[x]$. For $n \geqslant n_0$ with $n_0 \geqslant 1$ and suitable parameters κ and N, we will reduce this problem to a multiplication problem in $\mathbb{F}_{\pi^\kappa,t}[z]_N^{\circledcirc}$. If $n < n_0$, then we use a naive multiplication algorithm. The threshold n_0 is a sufficiently large absolute constant such that various asymptotic inequalities encountered during the proof hold for all $n \geqslant n_0$. The other parameters are determined as follows:

- $\epsilon := 1/1154$.
- $\ell := 3^{\lceil \log_3(n^\epsilon) \rceil}$ is the smallest power of three with $\ell \geqslant n^\epsilon$.
- With the notations from section 5.3 and for each $\nu > 0$, let $p_\nu = 2 q_\nu \ell + 1$ be the ν-th element in the list $P_1^*(1, 2\ell), P_2^*(1, 2\ell), P_3^*(1, 2\ell), \ldots$ with $\gcd(q_\nu, 6) = 1$.
- $d := \min \{ d \in \mathbb{N} : \ell^4 p_1 \cdots p_d \geqslant n \}$.
- $\kappa := \operatorname{lcm}(p_1 - 1, \ldots, p_d - 1)$.
- $s := 2 \cdot 3^{\lceil \log_3(2n/(\kappa p_1 \cdots p_d)) \rceil}$ is smallest in $2 \cdot 3^{\mathbb{N}}$ with $\kappa s p_1 \cdots p_d \geqslant 4n$.
- $N := s p_1 \cdots p_d$.

Since $\ell^4 p_1 \cdots p_{1150} \geqslant \ell^{1154} = \ell^{1/\epsilon} \geqslant n$, we must have $d \leqslant 1150$. From Lemma 5.4, we get

$$\ell < p_1 < \cdots < p_d \leqslant P_{d+2}^*(1, 2\ell) < \bar{C} (2\ell)^{\bar{L}} \leqslant \ell^{1+\epsilon},$$

for sufficiently large n, where

$$\bar{L} := 1 + (L+1)^{d+1} (L-1) < 1 + (2 + 2^{-1162})^{1151} 2^{-1162} < 1 + \epsilon,$$

and $\bar{C}$ is a constant. As before, the prime numbers $p_1, \ldots, p_d$ and the rest of the parameters can be computed in linear time $O(n)$. Let us proceed with a few useful bounds that hold for sufficiently large n. From the definitions of s and N, we clearly have

$$4n \leqslant \kappa N \leqslant 12n.$$

From $p_d < \ell^{1+\epsilon}$ and $\kappa = 2\ell \operatorname{lcm}(q_1, \ldots, q_d) \leqslant \ell^{1-d} p_1 \cdots p_d$, we get

$$\ell \leqslant \kappa \leqslant \ell^{1+d\epsilon} \leqslant \ell^2$$

and

$$\ell \leqslant 4\ell^{2-(d+1)\epsilon} = 4\ell^{4-(1+\epsilon)-(1+d\epsilon)} \leqslant \frac{4\ell^{4-(1+\epsilon)}}{\kappa} \leqslant \frac{4n}{\kappa p_1 \cdots p_d} \leqslant s \leqslant \frac{12n}{\kappa p_1 \cdots p_d} \leqslant \frac{12\ell^4}{\kappa} \leqslant 12\ell^3.$$

Step 0, continued: setting up the FFT field. As in the previous section, setting $\nu := p_1 \cdots p_d$, the defining polynomial μ for $\mathbb{F}_{\pi^\kappa}$ and a primitive ν-th root of unity ω can both be computed in time $O(n)$.

Step 1: Kronecker segmentation. Recall that $\kappa N \geqslant 4n$. In order to multiply two polynomials $a, b \in \mathbb{F}_{\pi,t}[x]$ of degree $< n$, we cut them in chunks $a_k, b_k \in \mathbb{F}_{\pi,t}[x]$ of degree $< \kappa/2$, i.e.

$$a = \sum_{k=0}^{N/2-1} a_k x^{(\kappa/2)k}, \qquad b = \sum_{k=0}^{N/2-1} b_k x^{(\kappa/2)k},$$

and form the polynomials

$$A = \sum_{k=0}^{N/2-1} a_k z^k, \qquad B = \sum_{k=0}^{N/2-1} b_k z^k.$$

Let $\bar{A}, \bar{B} \in \mathbb{F}_{\pi^\kappa,t}[z]_N^\circledcirc$ be the reductions of these polynomials modulo μ and $z^N + z^{N/2} + 1$. Since $\deg_z (A\,B) < N$ and $\deg_x (A\,B) < \kappa$, we can read off the product $A\,B$ from the product $\bar{A}\,\bar{B}$, after which $a\,b = (A\,B)(x^{\kappa/2})$. Computing $A, B, \bar{A}, \bar{B}$ from a and b clearly takes linear time $O(2^t n)$, and so does the retrieval of $A\,B$ and $a\,b$ from $\bar{A}\,\bar{B}$. In other words, we have shown that

$$\mathsf{M}_{\pi,t}(n) \leqslant \mathsf{M}^\circledcirc_{\pi^\kappa,t}(N) + O(2^t n). \tag{8.1}$$

Step 2: CRT transforms. At a second stage, we use tricyclic CRT transforms followed by traditional CRT transforms in order to reduce multiplication in $\mathbb{F}_{\pi^\kappa,t}[z]_N^\circledcirc$ to multiplication in $\mathbb{A}_t[\boldsymbol{x}]_{\boldsymbol{p}}^\circ$, where $\boldsymbol{x} = (x_1, \ldots, x_d)$, $\boldsymbol{p} = (p_1, \ldots, p_d)$, and $\mathbb{A} = \mathbb{F}_{\pi^\kappa}[u]_s^\circledcirc$:

$$\begin{aligned} \mathsf{M}^\circledcirc_{\pi^\kappa,t}(N) &\leqslant \mathsf{M}^\circ_{\mathbb{A},t}(p_1 \cdots p_d) + O(2^t N \lg N \kappa) \\ \mathsf{M}^\circ_{\mathbb{A},t}(p_1 \cdots p_d) &\leqslant \mathsf{M}^\circ_{\mathbb{A},t}(\boldsymbol{p}) + O(2^t |\boldsymbol{p}| \lg |\boldsymbol{p}| s\,\kappa). \end{aligned}$$

Since $O(|\boldsymbol{p}| \lg |\boldsymbol{p}| s\,\kappa) = O(N \lg |\boldsymbol{p}| \kappa) = O(N \lg N \kappa) = O(n \lg n)$, this yields

$$\mathsf{M}^\circledcirc_{\pi^\kappa,t}(N) \leqslant \mathsf{M}^\circ_{\mathbb{A},t}(\boldsymbol{p}) + O(2^t n \lg n).$$

We perform the multivariate cyclic convolutions in $\mathbb{A}_t[\boldsymbol{x}]_{\boldsymbol{p}}^\circ$ using fast Fourier multiplication, where the discrete Fourier transforms are computed with respect to $\boldsymbol{\omega} = (\omega^{\nu/p_1}, \ldots, \omega^{\nu/p_d})$. Since the only possible divisions of elements in $\mathbb{A}$ by integers are divisions by one, they come for free, so (3.8) yields

$$\mathsf{M}^\circ_{\mathbb{A},t}(\boldsymbol{p}) \leqslant 3\,\mathsf{F}_{\mathbb{A},t}(\boldsymbol{p}) + |\boldsymbol{p}|\,\mathsf{m}_{\mathbb{A},t} + O(2^t n \lg n).$$

The multivariate DFTs can be performed on the 2^t coefficients over $\mathbb{A}$ in parallel:

$$\mathsf{F}_{\mathbb{A},t}(\boldsymbol{p}) \leqslant 2^t\,\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + O(2^t n \lg n).$$

The reduction of a polynomial of degree $< 2\,\kappa$ in $\mathbb{F}_\pi[x]$ modulo μ using Barrett's algorithm [3] essentially requires two polynomial multiplications in $\mathbb{F}_\pi[x]$ of degree $< \kappa$, modulo the precomputation of a preinverse in time $O(\mathsf{M}_\pi(\kappa)) = O(n)$. Performing multiplications in $\mathbb{A}_t$ using Kronecker substitution, we thus have

$$\mathsf{m}_{\mathbb{A},t} \leqslant \mathsf{M}_{\pi,t}(2\,\kappa s) + 2 \cdot 2^t \cdot s\,\mathsf{M}_\pi(\kappa) + O(2^t \kappa s).$$

For sufficiently large n, we conclude that

$$\mathsf{M}^\circledcirc_{\pi^\kappa,t}(N) \leqslant 3 \cdot 2^t \cdot \mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) + |\boldsymbol{p}|\,\mathsf{M}_{\pi,t}(2\,\kappa s) + 2 \cdot 2^t \cdot |\boldsymbol{p}| s\,\mathsf{M}_\pi(\kappa) + O(2^t n \lg n). \tag{8.2}$$

Step 3: multivariate Rader reduction. This step is essentially the same as in section 7; setting $\boldsymbol{r_1} = \boldsymbol{p} - 1$, we obtain the bound

$$\mathsf{F}_{\mathbb{A}}(\boldsymbol{p}) \leqslant \mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{r_1}) + O(n \lg n). \tag{8.3}$$

Step 4: second CRT transforms. For $k = 1, \ldots, d$, we may decompose $r_{\boldsymbol{1},k} = p_k - 1 = 2\,\ell\,q_k$, where q_k is coprime with 6. Using multivariate CRT transforms, this yields

$$\mathsf{M}^\circ_{\mathbb{A}}(\boldsymbol{r_1}) \leqslant \mathsf{M}^\circ_{\mathbb{A}}(2q_1, \ldots, 2q_d, \ell, \ldots, \ell) + O(n \lg n).$$

Setting $\mathbb{B} = \mathbb{A}[\boldsymbol{y}]^{\circ}_{2\boldsymbol{q}}$ and $\boldsymbol{\ell} = (\ell, \overset{d\times}{\ldots}, \ell)$, this relation can be rewritten as

$$\mathsf{M}^{\circ}_{\mathbb{A}}(\boldsymbol{r_1}) \leqslant \mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) + O(n \lg n). \tag{8.4}$$

Step 5: Nussbaumer polynomial transforms. Since $\ell \mid (s/2)$, we have a fast root of unity $u^{3s/(2\ell)}$ of order ℓ in $\mathbb{A}$ and in $\mathbb{B}$. We may thus compute polycyclic convolutions of order $\boldsymbol{\ell}$ over $\mathbb{B}$ using fast Fourier multiplication, with triadic Nussbaumer polynomial transforms in the role of discrete Fourier transforms. Using that divisions by integers are again for free in $\mathbb{B}$, this yields

$$\begin{aligned}\mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\,\mathsf{m}_{\mathbb{B}} + O(|\boldsymbol{\ell}| \lg |\boldsymbol{\ell}|\,\mathsf{s}_{\mathbb{B}}) \\ &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}^{\circ}_{\mathbb{A}}(2\boldsymbol{q}) + O(n \lg n).\end{aligned}$$

Step 6: residual cyclic convolutions. Using CRT transforms, the residual multivariate cyclic convolutions of length $(2q_1, \ldots, 2q_d)$ are successively transformed into multivariate cyclic convolutions of lengths $(2, \ldots, 2, q_1, \ldots, q_d)$ and then $(2, \ldots, 2, q_1 \cdots q_d)$. Using that $s, q_1, \ldots, q_d$ are pairwise coprime, we get

$$\begin{aligned}\mathsf{M}^{\circ}_{\mathbb{A}}(2\boldsymbol{q}) &\leqslant \mathsf{M}^{\circ}_{\mathbb{A},d}(|\boldsymbol{q}|) + O(2^d \kappa s\,|\boldsymbol{q}| \lg |\boldsymbol{q}|) \\ &\leqslant \mathsf{M}^{\circ}_{\pi^\kappa,d}(s\,|\boldsymbol{q}|) + O(2^d \kappa s\,|\boldsymbol{q}| \lg(s\,|\boldsymbol{q}|)),\end{aligned}$$

whence

$$\begin{aligned}\mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}^{\circ}_{\mathbb{A}}(2\boldsymbol{q}) + O(n \lg n) \\ &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}^{\circ}_{\pi^\kappa,d}(s\,|\boldsymbol{q}|) + O(n \lg n).\end{aligned}$$

We perform the univariate cyclic products using Kronecker substitution. Since reductions of polynomials in $\mathbb{F}_\pi[x]_{2\kappa}$ modulo μ can be done in time $2\,\mathsf{M}_\pi(\kappa) + O(\kappa)$, this yields

$$\begin{aligned}\mathsf{M}^{\circ}_{\mathbb{B}}(\boldsymbol{\ell}) &\leqslant |\boldsymbol{\ell}|\,\mathsf{M}_{\pi,d}(2\kappa s\,|\boldsymbol{q}|) + 2\,|\boldsymbol{\ell}|\,|2\boldsymbol{q}|\,s\,\mathsf{M}_\pi(\kappa) + O(n \lg n) \\ &\leqslant 2N\kappa \frac{\mathsf{M}_{\pi,d}(2\kappa s\,|\boldsymbol{q}|)}{2\kappa s\,|2\boldsymbol{q}|} + 2\,|\boldsymbol{p}|\,s\,\mathsf{M}_\pi(\kappa) + O(n \lg n).\end{aligned} \tag{8.5}$$

Step 7: recurse. Combining the relations (8.1), (8.2), (8.3), (8.4) and (8.5), while using the fact that $6N\kappa \leqslant 72n$, we obtain

$$\frac{\mathsf{M}_{\pi,t}(n)}{2^t} \leqslant 72n \left(\frac{\mathsf{M}_{\pi,d}(2\kappa s\,|\boldsymbol{q}|)}{2\kappa s\,|\boldsymbol{q}|\,2^d} \right) + |\boldsymbol{p}| \frac{\mathsf{M}_{\pi,t}(2\kappa s)}{2^t} + 8\,|\boldsymbol{p}|\,s\,\mathsf{M}_\pi(\kappa) + O(n \lg n).$$

In terms of the normalized cost functions

$$\begin{aligned}\bar{\mathsf{M}}_{\pi,t}(n) &= \max_{0 \leqslant i \leqslant t} \frac{\mathsf{M}_{\pi,i}(n)}{2^i} \\ \mathsf{M}^{*}_{\pi,t}(n) &= \max_{2 \leqslant k \leqslant n} \frac{\bar{\mathsf{M}}_{\pi,t}(k)}{k \log k}\end{aligned}$$

and $M = 2\kappa s\,|\boldsymbol{q}|$, the above relation implies

$$\begin{aligned}\frac{\bar{\mathsf{M}}_{\pi,t}(n)}{n \log n} &\leqslant 72\,\mathsf{M}^{*}_{\pi,d}(2\kappa s\,|\boldsymbol{q}|) \frac{\log(2\kappa s\,|\boldsymbol{q}|)}{\log n} \\ &\quad + 24\,\mathsf{M}^{*}_{\pi,t}(2\kappa s) \frac{\log(2\kappa s)}{\log n} + 96\,\mathsf{M}^{*}_{\pi,0}(\kappa) \frac{\log \kappa}{\log n} + O(1) \\ &\leqslant 192\,\mathsf{M}^{*}_{\pi,\max(t,d)}(M) \frac{\log(2\kappa s\,|\boldsymbol{q}|)}{\log n} + O(1),\end{aligned}$$

where we used the fact that $2|\boldsymbol{p}|\kappa s \leqslant 24n$. Taking $t_0 = 1150$, so that $d \leqslant t_0$, this yields

$$\frac{\bar{\mathsf{M}}_{\pi,t_0}(n)}{n\log n} \leqslant 192\,\mathsf{M}^*_{\pi,t_0}(M)\,\frac{\log(2\kappa s|\boldsymbol{q}|)}{\log n} + O(1).$$

Recall that $s \leqslant 12\,\ell^3$ and $q_k < \ell^\epsilon$ for $k = 1, \ldots, d$, whence $s|\boldsymbol{q}| < 12\,\ell^{3+d\epsilon} \leqslant 12\,\ell^{4-4\epsilon}$, whereas $\kappa \leqslant \ell^2$. Consequently,

$$\log(2\kappa s|\boldsymbol{q}|) \leqslant (6-4\epsilon)\,\epsilon\log n + O(1) \leqslant 6\epsilon\log n$$

for large n. For some suitable universal constant $B > 0$, this yields

$$\frac{\bar{\mathsf{M}}_{\pi,t_0}(n)}{n\log n} \leqslant 1152\,\mathsf{M}^*_{\pi,t_0}(M)\,\epsilon + B,$$

For sufficiently large $n \geqslant n_0$, we have $M = 2\kappa s|\boldsymbol{q}| \leqslant 24\frac{|\boldsymbol{q}|}{|\boldsymbol{p}|}n \leqslant 24\,n\,\ell^{-d} < n$ and $1152\,\epsilon < 1-\epsilon$, whence

$$\frac{\bar{\mathsf{M}}_{\pi,t_0}(n)}{n\log n} \leqslant (1-\epsilon)\,\mathsf{M}^*_{\pi,t_0}(M) + B.$$

In a similar way as in the previous sections, this implies $\mathsf{M}^*_{\pi,t_0}(n) \leqslant \mathsf{M}^*_{\pi,t_0}(n_0) + B/\epsilon$ for all $n \geqslant n_0$, whence $\mathsf{M}_\pi(n) \leqslant \mathsf{M}^*_{\pi,0}(n)\,n\log n \leqslant \mathsf{M}^*_{\pi,t_0}(n)\,n\log n \leqslant O(n\lg n)$.

Remark 8.1. Since it suffices to consider the case when $t \leqslant t_0 = 1150$ is bounded in our proof, we can pay ourselves the luxury to use a naive algorithm for polynomial multiplications over $\mathbb{F}_{\pi,t}$ of bounded degree $\leqslant n_0$. It is an interesting question how to do such multiplications more efficiently. First of all, we notice that

$$\mathbb{F}_\pi[\theta_1, \ldots, \theta_t]/(\theta_1^2 - 1, \ldots, \theta_t^2 - 1) \cong \mathbb{F}_\pi[z_1, \ldots, z_t]/(z_1^2, \ldots, z_t^2),$$

via the the change of variables $\theta_1 = z_1 + 1, \ldots, \theta_t = z_t + 1$. The bound (3.2) implies that this change of variables can be performed in time $O(t2^t)$. In [50], Schost designed an efficient algorithm for multiplication in the truncated power series ring $\mathbb{F}_\pi[z_1, \ldots, z_t]/(z_1^2, \ldots, z_t^2)$. The idea is to embed this ring inside

$$\mathbb{F}_\pi[[\varepsilon]][z_1, \ldots, z_t]/(z_1^2 - \varepsilon z_1, \ldots, z_t^2 - \varepsilon z_t)$$

and then to use multipoint evaluation and interpolation at $z_1, \ldots, z_t \in \{0, \varepsilon\}$. It turns out that it suffices to compute with series of precision $O(\varepsilon^{t+1})$ in order to obtain the desired product. Altogether, this leads to an algorithm of bit complexity $O(t^2 2^t)$.

9. VARIANTS OF THEOREM 1.2

There are several variants and generalizations of Theorem 1.2 along similar lines as the results in [27, section 8]. First of all, we have the following:

THEOREM 9.1. *Assume that there exists a Linnik constant with $L < 1 + 2^{-1162}$ and let $m > 0$. Then the bit complexity $\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}(n)$ of multiplying two polynomials in $(\mathbb{Z}/m\,\mathbb{Z})[x]_n$ satisfies*

$$\mathsf{M}_{\mathbb{Z}/m\mathbb{Z}}(n) = O(n\log m\log(n\log m)), \qquad \textit{uniformly in } m.$$

Proof. Routine adaptation of the proofs of [27, Theorems 8.2 and 8.3]. □

Until now, we have systematically worked in the bit complexity model for implementations on a multitape Turing machine. But it is a matter of routine to adapt Theorem 1.2 to other complexity models. In the straight-line program (SLP) model, the most straightforward adaptation of [27, Theorem 8.4] goes as follows:

THEOREM 9.2. *Assume that there exists a Linnik constant with $L < 1 + 2^{-1162}$. Let $\mathcal{A}$ be an $\mathbb{F}_\pi$-algebra, where π is prime. We may multiply two polynomials in $\mathcal{A}[X]$ of degree less than n using $O(n \log n)$ additions, subtractions, and scalar multiplications, and $O(n (\log n)^{\log 24/\log 572})$ non-scalar multiplications. These bounds are uniform over all primes π and all $\mathbb{F}_\pi$-algebras $\mathcal{A}$.*

Proof. With respect to the proof of [27, Theorem 8.4], the only non-trivial thing to check concerns the number of non-scalar multiplications. These are due to the innermost multiplications of our recursive FFT-multiplications, when unrolling the recursion. Assume first that $\pi \neq 2$. Then one level of our recursion reduces the multiplication of two polynomials in $\mathbb{F}_\pi[x]_n$ to N multiplications in $\mathbb{F}_{\pi^\kappa}$, whence to $2N$ multiplications in $\mathbb{F}_\pi[x]_\kappa$. By construction, we have $\kappa \leqslant \ell^2 \leqslant (2n)^{2\epsilon}$ and $\kappa N \leqslant 8n$. Each recursive step therefore expands the data size by a factor at most 16 while passing from degree n to degree roughly $n^{2\epsilon} = n^{1/385}$. The algorithm therefore requires $\log\log n / \log 385 + O(1)$ recursive steps and $O(n (\log n)^{\log 16/\log 385})$ non-scalar multiplications. In the case when $\pi = 2$, the reasoning is similar: this time, the data size is expanded by a factor at most 24 at each recursive step (since $\kappa N \leqslant 12n$), whereas the degree in x descends from n to at most $n^{2\epsilon} = n^{1/572}$. This leads to the announced complexity bound. □

We notice that the number of non-scalar multiplications is far larger than $O(n 4^{\log^* n})$, as for the algorithm from [27, Theorem 8.4]. Ideally speaking, we would like to take κ exponentially smaller than n, which requires a sharper bound on $P^*_\nu(k)$ than the one from Lemma 5.4. Assuming that this is possible (modulo stronger number theoretic assumptions), the number of recursive steps would go down to $\log^* n + O(1)$, and the number of non-scalar multiplications to $O(n 24^{\log^* n})$. With more work, it is plausible that the constant 24 can be further reduced.

It is also interesting to observe that the bulk of the other $\mathbb{F}_\pi$-vector space operations are additions and subtractions: for some constant α with $0 < \alpha < 1$, the algorithm only uses $O(n (\log n)^\alpha)$ scalar multiplications. In a similar way as above, this number goes down to $O(n 2^{O(\log^* n)})$ modulo stronger number theoretic assumptions. For bounded $\pi = O(1)$, we also notice that scalar multiplications theoretically reduce to $\lg \pi = O(1)$ additions.

APPENDIX A. TURING MACHINE IMPLEMENTATIONS

Recall that all computations in this paper are done in the deterministic multitape Turing model [16]. In this appendix, we briefly review the cost of various basic types of data rearrangements when working in this model.

A.1. Arrays and sorting

Let R be a data type whose instances take s_R bits. An $n_1 \times \cdots \times n_d$ array $M_{i_1,\ldots,i_d}$ of elements in R is stored as a linear array of $n_1 \cdots n_d \mathsf{s}_R$ bits. We generally assume that the elements are ordered lexicographically by $(i_1, \ldots, i_d)$.

What is significant from a complexity point of view is that occasionally we must switch representations, to access an array (say 2-dimensional) by "rows" or by "columns". In the Turing model, we may transpose an $n_1 \times n_2$ matrix of elements in R in time

$$\mathsf{T}_R(n_1, n_2) \;=\; O(\mathsf{s}_R n_1 n_2 \lg \min(n_1, n_2)),$$

using the algorithm of [5, appendix]. Briefly, the idea is to split the matrix into two halves along the "short" dimension, and transpose each half recursively.

We will also require more complex rearrangements of data, for which we resort to sorting. Suppose that we can compare elements of R in linear time $O(\mathsf{s}_R)$. Then an array of n elements of R may be sorted in time

$$\mathsf{P}_R(n) \;=\; O(\mathsf{s}_R n \lg n),$$

using merge sort [36], which can be implemented efficiently on a Turing machine.

Let us now consider a general permutation $\sigma \in \mathfrak{S}_n$. We assume that σ is represented by the array $(\sigma(1), \ldots, \sigma(n))$, which takes space $O(n \lg n)$ on a Turing tape. Given a general data type R, the permutation naturally acts on arrays of length with entries in R. As long as $\lg n = O(\mathsf{s}_R)$, this action can be computed in time $O(\mathsf{s}_R n \lg n)$, by tagging the i-th entry of the array by $\sigma(i)$, sorting the tagged entries, and then removing the tags.

Unfortunately, for some of the data rearrangements in this paper, we do not have $\lg n = O(\mathsf{s}_R)$. Nevertheless, certain special kinds of permutations can still be performed in time $O(\mathsf{s}_R n \lg n)$. Transpositions are one important example. In the next subsections, we will consider a few other examples.

A.2. Basic multivariate rewritings

LEMMA A.1. *Consider a permutation $\sigma \in \mathfrak{S}_\ell$. Let $\boldsymbol{n} = (n_1, \ldots, n_\ell) \in (\mathbb{N}^>)^\ell$ and $\boldsymbol{m} = (m_1, \ldots, m_\ell) = (n_{\sigma(1)}, \ldots, n_{\sigma(\ell)})$. Then*

$$\mathsf{C}(R^{\boldsymbol{n}} \leftrightarrow R^{\boldsymbol{m}}) \;=\; O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|)$$

Proof. Let $C > 0$ be an absolute constant with $\mathsf{T}_R(k_1, k_2) \leqslant C\, \mathsf{s}_R k_1 k_2 \log \min(k_1, k_2)$ for all k_1, k_2. Let us prove by induction on ℓ that $\mathsf{C}(R^{\boldsymbol{n}} \leftrightarrow R^{\boldsymbol{m}}) \leqslant C\, \mathsf{s}_R |\boldsymbol{n}| \log |\boldsymbol{n}|$. For $\ell \leqslant 1$, the result is clear, so assume that $l > 1$. Let $\pi \colon \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ be the mapping with $\pi(1) = \sigma(1)$, $\pi(2) = 1, \ldots, \pi(\sigma(1)) = \sigma(1) - 1$ and $\pi(k) = k$ for $k > \sigma(1)$. Setting $\boldsymbol{n}' = (n_1, \ldots, n_{\sigma(1)})$, $\boldsymbol{r}' = (n_{\pi(1)}, \ldots, n_{\pi(\sigma(1))})$ and $\boldsymbol{r} = (n_{\pi(1)}, \ldots, n_{\pi(\ell)})$, the natural isomorphism between $R^{\boldsymbol{n}'}$ and $R^{\boldsymbol{r}'}$ corresponds to the transposition of an $(n_1 \cdots n_{\sigma(1)-1}) \times n_{\sigma(1)}$ matrix with coefficients in R. Since conversions between $R^{\boldsymbol{n}}$ and $R^{\boldsymbol{r}}$ amount to $n_{\sigma(1)+1} \cdots n_\ell$ such transpositions, it follows that

$$\mathsf{C}(R^{\boldsymbol{n}} \leftrightarrow R^{\boldsymbol{r}}) \;\leqslant\; C\, \mathsf{s}_R |\boldsymbol{n}| \log n_{\sigma(1)}.$$

By the induction hypothesis, we also have

$$\mathsf{C}(R^{\boldsymbol{r}} \leftrightarrow R^{\boldsymbol{m}}) \;\leqslant\; C\, \mathsf{s}_{R^{\sigma(1)}} \frac{|\boldsymbol{n}|}{n_{\sigma(1)}} \log \frac{|\boldsymbol{n}|}{n_{\sigma(1)}} \;=\; C\, \mathsf{s}_R |\boldsymbol{n}| \log \frac{|\boldsymbol{n}|}{n_{\sigma(1)}},$$

whence

$$\mathsf{C}(R^{\boldsymbol{n}} \leftrightarrow R^{\boldsymbol{m}}) \;\leqslant\; \mathsf{C}(R^{\boldsymbol{n}} \leftrightarrow R^{\boldsymbol{r}}) + \mathsf{C}(R^{\boldsymbol{r}} \leftrightarrow R^{\boldsymbol{m}}) \;\leqslant\; C\, \mathsf{s}_R |\boldsymbol{n}| \log |\boldsymbol{n}|,$$

as desired. □

COROLLARY A.2. *For any monic polynomials $P_1 \in R[x_1], \ldots, P_\ell \in R[x_\ell]$, the conversions*

$$R[x_1]/(P_1) \cdots [x_\ell]/(P_\ell) \leftrightarrow R[x_{\sigma(1)}]/(P_{\sigma(1)}) \cdots [x_{\sigma(\ell)}]/(P_{\sigma(\ell)})$$

can be performed in time $O(\mathsf{s}_R n \lg n)$, where $n = \deg P_1 \cdots \deg P_\ell$.

Proof. Setting $\boldsymbol{n} = (\deg P_1, \ldots, \deg P_\ell)$, we may naturally represent elements in

$$R[x_1]/(P_1) \cdots [x_\ell]/(P_\ell)$$

by ℓ-variate arrays in $R^{\boldsymbol{n}}$. Similarly, elements of

$$R[x_{\sigma(1)}]/(P_{\sigma(1)}) \cdots [x_{\sigma(\ell)}]/(P_{\sigma(\ell)})$$

are represented by arrays in $R^{\boldsymbol{m}} = (n_{\sigma(1)}, \ldots, n_{\sigma(\ell)})$. We now apply the lemma. □

COROLLARY A.3. *We have*

$$\mathsf{C}(R[\boldsymbol{x}]^\circ_{\boldsymbol{n}} \leftrightarrow R[\boldsymbol{y}]^\circ_{\boldsymbol{m}}) \;=\; O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|).$$

Proof. Take $P_i = x_i^{n_i} - 1$ for $i = 1, \ldots, \ell$. □

A.3. CRT transforms

We start with two lemmas from [23, section 2.4]. For convenience we reproduce the proofs here.

LEMMA A.4. *Let $m_1, m_2 \geqslant 2$ be relatively prime and $n = m_1 m_2$. Then*

$$\mathsf{C}(R[x]^\circ_n \leftrightarrow R[x_1]^\circ_{m_1}[x_2]^\circ_{m_2}) \;=\; O(\mathsf{s}_R n \lg \min(m_1, m_2)).$$

Proof. Let $c = m_2^{-1} \bmod m_1$, and let

$$\beta\colon R[x]/(x^{m_1 m_2} - 1) \to R[x_1, x_2]/(x_1^{m_1} - 1, x_2^{m_2} - 1)$$

denote the homomorphism that maps x to $x_1^c x_2$, and acts as the identity on R. Suppose that we wish to compute $\beta(F)$ for some input polynomial

$$F = \sum_{k=0}^{m_1 m_2 - 1} F_k x^k \in R[x]/(x^{m_1 m_2} - 1).$$

Interpreting the list $(F_0, \ldots, F_{m_1 m_2 - 1})$ as an $m_1 \times m_2$ array, the (i_1, i_2)-th entry corresponds to $F_{i_1 m_2 + i_2}$. After transposing the array, which costs $O(\mathsf{s}_R m_1 m_2 \lg \min(m_1, m_2))$ bit operations, we have an $m_2 \times m_1$ array, whose (i_2, i_1)-th entry is $F_{i_1 m_2 + i_2}$. Now for each i_2, cyclically permute the i_2-th row by $(i_2 c \bmod m_1)$ slots; altogether this uses only $O(\mathsf{s}_R m_1 m_2)$ bit operations. The result is an $m_2 \times m_1$ array whose (i_2, i_1)-th entry is $F_{(i_1 - i_2 c \bmod m_1) m_2 + i_2}$, which is exactly the coefficient of

$$x_1^{((i_1 - i_2 c) m_2 + i_2) c} x_2^{(i_1 - i_2 c) m_2 + i_2} = x_1^{i_1} x_2^{i_2}$$

in $\beta(F)$. The inverse map β^{-1} may be computed by reversing this procedure. □

LEMMA A.5. *Let $n = m_1 \cdots m_\ell$ where the m_i are pairwise coprime. Then*

$$\mathsf{C}(R[x]^\circ_n \leftrightarrow R[x_1]^\circ_{m_1} \cdots [x_k]^\circ_{m_\ell}) \;=\; O(\mathsf{s}_R n \lg n).$$

Proof. Using Lemma A.4, we construct a sequence of isomorphisms

$$\begin{aligned} R[x]_n^\circ &\cong R[x_1]_{m_1}^\circ [w_2]_{m_2 \cdots m_\ell}^\circ \\ &\cong R[x_1]_{m_1}^\circ [x_1]_{m_2}^\circ [w_3]_{m_3 \cdots m_\ell}^\circ \\ &\;\;\vdots \\ &\cong R[x_1]_{m_1}^\circ \cdots [x_k]_{m_\ell}^\circ \end{aligned}$$

the i-th of which may be computed in $C \mathsf{s}_R n \log n_i$ bit operations for some universal constant $C > 0$. The overall cost is $\sum_i C \mathsf{s}_R n \log n_i = C \mathsf{s}_R n \log n$ bit operations. □

A.4. Multivariate CRT transforms

LEMMA A.6. *Let $\boldsymbol{n} = (n_1, \ldots, n_\ell) \in (\mathbb{N}^>)^\ell$ and let $n_i = m_{i,1} \cdots m_{i,k_i}$ be the prime power factorization of each n_i. Setting $\boldsymbol{m} = (m_{1,1}, \ldots, m_{1,k_1}, \ldots, m_{\ell,1}, \ldots, m_{\ell,k_\ell})$, we have*

$$\mathsf{C}(R[\boldsymbol{x}]_{\boldsymbol{n}}^\circ \leftrightarrow R[\boldsymbol{y}]_{\boldsymbol{m}}^\circ) \;=\; O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|).$$

Proof. We prove the assertion by induction over ℓ. For $\ell \leqslant 1$, the result follows from Lemma A.5, so assume that $l > 1$. Denote $\boldsymbol{n}' = (n_2, \ldots, n_\ell)$, $\boldsymbol{m}' = (m_{2,1}, \ldots, m_{2,k_2}, \ldots, m_{\ell,1}, \ldots, m_{\ell,k_\ell})$, and $\boldsymbol{m}^\sharp = (m_{1,1}, \ldots, m_{1,k_1})$. Lemma A.5 yields $\mathsf{C}(R[x_1]_{n_1}^\circ \leftrightarrow R[\boldsymbol{y}^\sharp]_{\boldsymbol{m}^\sharp}^\circ) = O(\mathsf{s}_R n_1 \lg n_1)$, whence

$$\mathsf{C}(R[\boldsymbol{x}]_{\boldsymbol{n}}^\circ \leftrightarrow R[\boldsymbol{y}^\sharp]_{\boldsymbol{m}^\sharp}^\circ [\boldsymbol{x}']_{\boldsymbol{n}'}^\circ) \;=\; O(\mathsf{s}_R |\boldsymbol{n}| \lg n_1).$$

The induction hypothesis also yields

$$\mathsf{C}(R[\boldsymbol{y}^\sharp]_{\boldsymbol{m}^\sharp}^\circ [\boldsymbol{x}']_{\boldsymbol{n}'}^\circ \leftrightarrow R[\boldsymbol{y}]_{\boldsymbol{m}}^\circ) \;=\; O\Big(\mathsf{s}_{R[\boldsymbol{y}^\sharp]_{\boldsymbol{m}^\sharp}^\circ} |\boldsymbol{n}'| \lg |\boldsymbol{n}'|\Big) = O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}'|).$$

Adding up the these two bounds, the result follows. □

Given $\boldsymbol{n} = (n_1, \ldots, n_\ell) \in (\mathbb{N}^>)^\ell$, we recall that the classification of finite abelian groups implies the existence of a unique tuple $\boldsymbol{n}' = (n_1', \ldots, n_{\ell'}')$ with $2 \leqslant n_{\ell'}' \mid n_{\ell'-1}' \mid \cdots \mid n_1'$, $\ell' \leqslant \ell$, $|\boldsymbol{n}'| = |\boldsymbol{n}|$, and

$$\mathbb{Z}/(n_1 \mathbb{Z}) \times \cdots \times \mathbb{Z}/(n_\ell \mathbb{Z}) \;\cong\; \mathbb{Z}/(n_1' \mathbb{Z}) \times \cdots \times \mathbb{Z}/(n_{\ell'}' \mathbb{Z}).$$

We call $\boldsymbol{n}'$ the normal form of $\boldsymbol{n}$ and say that $\boldsymbol{n}$ is in *normal form* if $\boldsymbol{n}' = \boldsymbol{n}$.

LEMMA A.7. *If $\boldsymbol{n}'$ is the normal form of $\boldsymbol{n}$, then*

$$\mathsf{C}(R[\boldsymbol{x}]_{\boldsymbol{n}}^\circ \leftrightarrow R[\boldsymbol{x}']_{\boldsymbol{n}'}^\circ) \;=\; O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|).$$

Proof. Let $\boldsymbol{m}$ and $\boldsymbol{m}'$ be the tuples obtained when performing the entry-wise prime power factorizations of $\boldsymbol{n}$ and $\boldsymbol{n}'$ as in Lemma A.6. From Lemma A.6 it follows that $\mathsf{C}(R[\boldsymbol{x}]_{\boldsymbol{n}}^\circ \leftrightarrow R[\boldsymbol{y}]_{\boldsymbol{m}}^\circ) = O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|)$ and $\mathsf{C}(R[\boldsymbol{x}']_{\boldsymbol{n}'}^\circ \leftrightarrow R[\boldsymbol{y}']_{\boldsymbol{m}'}^\circ) = O(\mathsf{s}_R |\boldsymbol{n}'| \lg |\boldsymbol{n}'|) = O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|)$. Since $\boldsymbol{m}'$ is a permutation of $\boldsymbol{m}$, the result follows from Lemma A.1. □

We say that two tuples $\boldsymbol{n} = (n_1, \ldots, n_\ell)$ and $\boldsymbol{n}' = (n_1', \ldots, n_{\ell'}')$ are *equivalent* if they admit the same normal form.

COROLLARY A.8. *If $\boldsymbol{n}$ and $\boldsymbol{n}'$ are equivalent, then*

$$\mathsf{C}(R[\boldsymbol{x}]_{\boldsymbol{n}}^\circ \leftrightarrow R[\boldsymbol{x}']_{\boldsymbol{n}'}^\circ) \;=\; O(\mathsf{s}_R |\boldsymbol{n}| \lg |\boldsymbol{n}|).$$

□

A.5. Tensor products of direct sums

Consider an R-algebra of the form $A = \bigoplus_{i \in \mathcal{I}} A_i$, where $\mathcal{I} = \{i^{[1]}, \ldots, i^{[r]}\}$ is finite and totally ordered $i^{[1]} < \cdots < i^{[r]}$. Elements $x \in A$ are of the form $x = \sum_{i \in \mathcal{I}} x_i$ with $x_i \in A_i$, so we may naturally represent them on a Turing tape by concatenating the representations of $x_{i^{[1]}}, \ldots, x_{i^{[r]}}$. In particular, if each A_i is an R-algebra of dimension s_i whose elements are represented by vectors of size s_i with entries in R, then elements in A are simply represented as vectors of size $\sum_{i \in \mathcal{I}} s_i$ with entries in R.

Now consider d such algebras $A_1 = \bigoplus_{i_1 \in \mathcal{I}_1} A_{1,i_1}, \ldots, A_d = \bigoplus_{i_d \in \mathcal{I}_d} A_{d,i_d}$ and let us form their tensor product $B = \bigotimes_{k=1}^{d} A_k$. Let s_{k,i_k} be the dimension of the R-subalgebra A_{k,i_k} for $k = 1, \ldots, d$ and $i_k \in \mathcal{I}_k$. Then elements in B are recursively represented as vectors of size $s_d = \sum_{i_d \in \mathcal{I}_d} s_{d,i_d}$ with entries in $B' = \bigotimes_{k=1}^{d-1} A_k$. Using the distributivity of $\otimes$ over $\oplus$, we have a natural isomorphism

$$B = \bigotimes_{k=1}^{d} A_k \cong \bigoplus_{\boldsymbol{i} \in \mathcal{I}} B_{\boldsymbol{i}},$$

where

$$\begin{aligned} \mathcal{I} &= \mathcal{I}_1 \times \cdots \times \mathcal{I}_d \\ B_{\boldsymbol{i}} = B_{(i_1, \ldots, i_d)} &= \bigotimes_{k=1}^{d} A_{k,i_k}. \end{aligned}$$

When endowing the index set $\mathcal{I}$ with the lexicographical ordering, this isomorphism requires some reordering of data on Turing tapes, but we can compute it reasonably efficiently as follows:

LEMMA A.9. *With the above notations, $t = s_1 \cdots s_d$, and $\delta = \max(|\mathcal{I}_1|, \ldots, |\mathcal{I}_d|)$, we have*

$$\mathsf{C}\left(\bigotimes_{1 \leqslant k \leqslant d} A_k \leftrightarrow \bigoplus_{\boldsymbol{i} \in \mathcal{I}} B_{\boldsymbol{i}} \right) = O(\mathsf{s}_R\, t\, d\, \delta).$$

Proof. Let us prove the assertion by induction on d. For $d = 1$, we have nothing to do, so assume that $d > 1$. We may consider an element y of B as a vector of size $s_2 \cdots s_d$ with coefficients in $A_1 \cong R^{s_1}$. For each $i_1 \in \mathcal{I}_1$, we may compute the projection x_{1,i_1} of $x_1 \in A_1$ on A_{1,i_1} in time $O(\mathsf{s}_R s_1)$. Doing this for all $s_2 \cdots s_d$ entries of y, we obtain an element y_{i_1} of

$$y_{i_1} \in A_{1,i_1} \otimes A_2 \otimes \cdots \otimes A_d.$$

This computation takes a time $O(\mathsf{s}_R t)$. Using the induction hypothesis, we can convert y_{i_1} into an element

$$z_{i_1} \in \bigoplus_{(i_2, \ldots, i_d) \in \mathcal{I}_2 \times \cdots \times \mathcal{I}_d} B_{i_1, \ldots, i_d}$$

in time $O(\mathsf{s}_R\, t\, (d-1)\, \delta\, s_{1,i_1} / s_1)$. Doing this in order for all $i_1 \in \mathcal{I}_1$ and concatenating the resulting z_{i_1}, we obtain the conversion of y as an element of $\bigoplus_{\boldsymbol{i} \in \mathcal{I}} B_{\boldsymbol{i}}$. The total computation time is bounded by

$$O\left(\sum_{i_1 \in \mathcal{I}_1} \mathsf{s}_R t + \mathsf{s}_R\, t\, (d-1)\, \delta \frac{s_{1,i_1}}{s_1} \right) = O(\mathsf{s}_R\, t\, |\mathcal{I}_1| + \mathsf{s}_R\, t\, (d-1)\, \delta) = O(\mathsf{s}_R\, t\, d\, \delta).$$

The conversion in the opposite direction can be computed in the same time by reversing the algorithm. □

BIBLIOGRAPHY

[1] R. Agarwal and J. Cooley. New algorithms for digital convolution. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(5):392–410, 1977.

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

[3] P. Barrett. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 311–323. Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.

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

[5] A. Bostan, P. Gaudry, and É. Schost. Linear recurrences with polynomial coefficients and application to integer factorization and Cartier-Manin operator. *SIAM J. Comput.*, 36:1777–1806, 2007.

[6] C. B. Boyer. *A History of Mathematics*. Princeton Univ. Press, First paperback edition, 1985.

[7] R. P. Brent and P. Zimmermann. *Modern Computer Arithmetic*. Cambridge University Press, 2010.

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

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

[10] S. Covanov and E. Thomé. Fast integer multiplication using generalized Fermat primes. *Math. Comp.*, 88:1449–1477, 2019.

[11] A. De, P. P. Kurur, C. Saha, and R. Saptharishi. Fast integer multiplication using modular arithmetic. *SIAM J. Comput.*, 42(2):685–699, 2013.

[12] M. Fürer. Faster integer multiplication. In *Proceedings of the Thirty-Ninth ACM Symposium on Theory of Computing, STOC 2007*, pages 57–66. New York, NY, USA, 2007. ACM Press.

[13] M. Fürer. Faster integer multiplication. *SIAM J. Comput.*, 39(3):979–1005, 2009.

[14] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, New York, 3rd edition, 2013.

[15] I. J. Good. The interaction algorithm and practical Fourier analysis. *Journal of the Royal Statistical Society*, Series B. 20(2):361–372, 1958.

[16] C. H.Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[17] D. Harvey. Faster arithmetic for number-theoretic transforms. *J. Symbolic Comput.*, 60:113–119, 2014.

[18] D. Harvey. Faster truncated integer multiplication. https://arxiv.org/abs/1703.00640, 2017.

[19] D. Harvey and J. van der Hoeven. Faster integer and polynomial multiplication using cyclotomic coefficient rings. Technical Report, ArXiv, 2017. http://arxiv.org/abs/1712.03693.

[20] D. Harvey and J. van der Hoeven. Faster integer multiplication using plain vanilla FFT primes. *Math. Comp.*, 88(315):501–514, 2019.

[21] D. Harvey and J. van der Hoeven. Faster integer multiplication using short lattice vectors. In R. Scheidler and J. Sorenson, editors, *Proc. of the 13-th Algorithmic Number Theory Symposium*, Open Book Series 2, pages 293–310. Mathematical Sciences Publishes, Berkeley, 2019.

[22] D. Harvey and J. van der Hoeven. Integer multiplication in time $O(n \log n)$. Technical Report, HAL, 2019. http://hal.archives-ouvertes.fr/hal-02070778.

[23] D. Harvey and J. van der Hoeven. Faster polynomial multiplication over finite fields using cyclotomic coefficient rings. Shortened version of [19]. Accepted for publication in J. of Complexity.

[24] D. Harvey, J. van der Hoeven, and G. Lecerf. Faster polynomial multiplication over finite fields. Technical Report, ArXiv, 2014. http://arxiv.org/abs/1407.3361. Preprint version of [27].

[25] D. Harvey, J. van der Hoeven, and G. Lecerf. Even faster integer multiplication. *Journal of Complexity*, 36:1–30, 2016.

[26] D. Harvey, J. van der Hoeven, and G. Lecerf. Fast polynomial multiplication over $F_{2^{60}}$. In *Proc. ISSAC '16*, pages 255–262. New York, NY, USA, 2016. ACM.

[27] D. Harvey, J. van der Hoeven, and G. Lecerf. Faster polynomial multiplication over finite fields. *J. ACM*, 63(6), 2017. Article 52.

[28] D. R. Heath-Brown. Zero-free regions for Dirichlet L-functions, and the least prime in an arithmetic progression. *Proc. London Math. Soc.*, 64(3):265–338, 1992.

[29] M. T. Heideman, D. H. Johnson, and C. S. Burrus. Gauss and the history of the fast Fourier transform. *Arch. Hist. Exact Sci.*, 34(3):265–277, 1985.

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

[31] J. van der Hoeven. Faster Chinese remaindering. Technical Report, HAL, 2016. http://hal.archives-ouvertes.fr/hal-01403810.

[32] J. van der Hoeven, R. Larrieu, and G. Lecerf. Implementing fast carryless multiplication. In J. Blömer, I. S. Kotsireas, T. Kutsia, and D. E. Simos, editors, *Proc. MACIS 2017, Vienna, Austria*, Lect. Notes in Computer Science, pages 121–136. Cham, 2017. Springer International Publishing.

[33] A. A. Karatsuba. The complexity of computations. *Proc. of the Steklov Inst. of Math.*, 211:169–183, 1995. English translation; Russian original at pages 186–202.

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

[35] D. E. Knuth. *The Art of Computer Programming*, volume 2: Seminumerical Algorithms. Addison-Wesley, 1969.

[36] D. E. Knuth. *The Art of Computer Programming*, volume 3: Sorting and Searching. Addison-Wesley, Reading, MA, 1998.

[37] S. Lang. *Algebra*. Springer Verlag, 2002.

[38] J. Li, K. Pratt, and G. Shakan. A lower bound for the least prime in an arithmetic progression. *The Quarterly Journal of Mathematics*, 68(3):729–758, 2017.

[39] Yu. V. Linnik. On the least prime in an arithmetic progression I. The basic theorem. *Rec. Math. (Mat. Sbornik) N.S.*, 15(57):139–178, 1944.

[40] Yu. V. Linnik. On the least prime in an arithmetic progression II. The Deuring-Heilbronn phenomenon. *Rec. Math. (Mat. Sbornik) N.S.*, 15(57):347–368, 1944.

[41] O. Neugebauer. *The Exact Sciences in Antiquity*. Brown Univ. Press, Providence, R.I., 1957.

[42] H. J. Nussbaumer. Fast polynomial transform algorithms for digital convolution. *IEEE Trans. Acoust. Speech Signal Process.*, 28(2):205–215, 1980.

[43] H. J. Nussbaumer. *Fast Fourier Transforms and Convolution Algorithms*. Springer-Verlag, 2nd edition, 1982.

[44] H. J. Nussbaumer and P. Quandalle. Computation of convolutions and discrete Fourier transforms by polynomial transforms. *IBM J. Res. Develop.*, 22(2):134–144, 1978.

[45] J. M. Pollard. The fast Fourier transform in a finite field. *Mathematics of Computation*, 25(114):365–374, 1971.

[46] C. M. Rader. Discrete Fourier transforms when the number of data samples is prime. *Proc. IEEE*, 56:1107–1108, 1968.

[47] A. Schönhage. Multiplikation großer Zahlen. *Computing*, 1(3):182–196, 1966.

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

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

[50] É. Schost. Multivariate power series multiplication. In Manuel Kauers, editor, *ISSAC '05*, pages 293–300. New York, NY, USA, 2005. ACM Press.

[51] V. Shoup. New algorithms for finding irreducible polynomials over finite fields. *Math. Comp.*, 189:435–447, 1990.

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

[53] I. Shparlinski. On finding primitive roots in finite fields. *Theoret. Comput. Sci.*, 157(2):273–275, 1996.

[54] D. E. Smith. *History of Mathematics*, volume 2. Dover, 1958.

[55] L. H. Thomas. Using computers to solve problems in physics. *Applications of digital computers*, 458:42–57, 1963.

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

[57] T. Xylouris. On the least prime in an arithmetic progression and estimates for the zeros of Dirichlet L-functions. *Acta Arith.*, 1:65–91, 2011.