Univariate Polynomials: Nearly Optimal Algorithms for Factorization and Rootfinding

Victor Y. Pan*
Mathematics and Computer Science Department
Lehman College, CUNY, Bronx, NY 10468
vpan@alpha.lehman.cuny.edu

Summary

To approximate all roots (zeros) of a univariate polynomial, we develop two effective algorithms and combine them in a single recursive process. One algorithm computes a basic well isolated zero-free annulus on the complex plane, whereas another algorithm numerically splits the input polynomial of the n-th degree into two factors balanced in the degrees and with the zero sets separated by the basic annulus. Recursive combination of the two algorithms leads to recursive computation of the complete numerical factorization of a polynomial into the product of linear factors and further to the approximation of the roots. The new rootfinder incorporates the earlier techniques of Schönhage and Kirrinnis and our old and new techniques and yields nearly optimal (up to polylogarithmic factors) arithmetic and Boolean cost estimates for the complexity of both complete factorization and rootfinding. The improvement over our previous record Boolean complexity estimates is by roughly the factor of n for complete factorization and also for the approximation of well-conditioned (well isolated) roots, whereas the same algorithm is also optimal (under both arithmetic and Boolean models of computing) for the worst case input polynomial, where the roots can be ill-conditioned, forming clusters. (The worst case bounds are supported by our previous algorithms as well.) All our algorithms allow processor efficient acceleration to achieve solution in polylogarithmic parallel time.

Keywords

Padé approximation, Graeffe's lifting, univariate polynomials, rootfinding, numerical polynomial factorization, geometry of polynomial zeros, computational complexity

2000 AMS Math. Subject Classification

68Q40, 68Q25, 65D99, 12Y05

*Supported by NSF Grant CCR 9732206 and PSC-CUNY Award 669363, 61393-0030, and 62435-0031

ISSAC 2001, 7/01, Ontario, Canada
©2001 ACM 1-58113-417-7/ 01/ 0007 $5.00

Introduction

Numerical rootfinding for a univariate polynomial is a classical problem four millennea old, which for many centuries remained most influential for the development of mathematics (cf. [29]). This is still a major research topic with highly important applications to computer algebra, in particular to the solution of polynomial systems of equations [2], [16], [29], [20], [21] (cf. also [30], [34], and [35] on the applications to the computation of approximate polynomial gcds and to the algebraic eigenproblem). The rootfinder in [26], [28] relies on recursive numerical splitting of the input polynomial $p(x)$ into the product of smaller degree (and ultimately linear) factors and reaches optimal (up to polylogarithmic factors) bounds on the asymptotic arithmetic and Boolean time of rootfinding for the worst case input polynomial. This case covers polynomials with ill-conditioned (clustered) zeros, which typically represent polynomials with multiple zeros after the numerical truncation of the coefficients. The bounds on the computational precision and the Boolean cost of the algorithm, however, are too high (off by the factor of n, the degree of $p(x)$, from the information lower bounds) at the auxiliary stage of polynomial factorization (this stage is of independent interest as well) and for the practically important case of rootfinding for the input polynomials with well-conditioned (isolated) zeros.

In the present work, we include an improved block of lifting/descending, which simplifies substantially (by the order of magnitude) the subsequent stage of splitting a polynomial into the product of two factors. At the latter stage, we also apply a new improved algorithm. Furthermore, we simplify the construction in [26], [28] at the stage of the factorization of a higher order derivative of the input polynomial. With the improved algorithm, we keep optimality (up to polylog factors) of the rootfinding for the worst case input polynomial but simultaneously reach it in the case of well-conditioned zeros as well as for the complete factorization of a polynomial. Technical statement of these results is in section 1 of Part II.

All algorithms of this paper allow work (processor) efficient parallel acceleration. This yields polylogarithmic parallel time bounds preserving work (processor) optimality. Apparently, our bit-operation cost bounds can be slightly improved, by roughly logarithmic factor, if one applies fast integer arithmetic based on the binary segmentation techniques (cf. [4, section 3.3], [15], [38], and [39]). These techniques are slightly superior to the FFT-based arithmetic, on which we rely in our estimates. Further comparision with some related works will be given in the appendix (see also [5]).

Our algorithms involve several advanced techniques (some of independent interest, e.g., reversed Graeffe's lifting incorporating Padé approximation) and generally require computations with multiple precision. Substantial further work is required to implement them effectively and make practically competitive. The implementation should be much simpler for the algorithms of section 3 of Part II, which supply the basis for effective splitting for a large class of input polynomials.

Our rootfinder remains nearly optimal (up to polylogarithmic factors) even for the more limited tasks of approximating a single root or a few roots of a polynomial, but in this case the computational cost is slightly lower and the implementation is simpler in distinct approaches, which use no splitting [25], [31].

The presentation of our rootfinder was challenging because it is bulit at the top of several highly developed constructions and incorporates and improves their advanced techniques. (Already [39] has 72 pages and [15] has 67 pages.) This ruled out a self-contained presentation and required inclusion of several citations. (We included a more complete exposition into [5].) Furthermore, the statement of the problem and the final complexity results are relatively compact but the techniques supporting our rootfinder cannot be unified. At least two groups of very different techniques are inviolved. We partition our paper respectively into two parts, each with separate enumeration of its sections, equations, theorems etc. In Part I we describe splitting algorithms. In Part II we combine the splitting results of Part I with the search for the basic annuli in a recursive process of nearly optimal rootfinding and state the resulting complexity estimates. Some auxiliary materials for Part I and comparison with some related works are presented in the appendix.

Acknowledgement

I am grateful to the referees for helpful comments.

Part I: Preconditioned Splitting into Factors

1. SPLITTING THEOREMS AND ORGANIZATION OF PART I

We begin with some definitions.

$$p = p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{j=1}^{n} (x - z_j), \quad p_n \neq 0, \qquad (1.1)$$

$$A = A(X, r_-, r^+) = \left\{ x : r_- \leq |x - X| \leq r^+ \right\}, \qquad (1.2)$$

$$|u| = \|u(x)\| = \sum_i |u_i| \text{ for } u = u(x) = \sum_i u_i x^i, \qquad (1.3)$$

$$\mu(b) = O((b \log b) \log\log b), \qquad (1.4)$$

$\mu(b)$ bit-operations suffice to multiply two integers modulo $2^b + 1$. A polynomial p is given with its coefficients; w.l.o.g. [15], [28], let all its zeros satisfy the bounds

$$|z_j| \leq 1, \quad j = 1, \ldots, n. \qquad (1.5)$$

"op" is an arithmetic operation, a comparison of two real numbers, or the computation of the values $|z|$ and $|z|^{1/k}$ for a complex z and a positive integer k. $\psi = r^+/r_-$ is the *relative width* of the annulus A in (1.2) and the *isolation ratio* of its internal disc

$$D = D(X, r_-) = \{x : |x| \leq r_-\}; \qquad (1.6)$$

this disc is called ψ*-isolated* [25], [29]. Hereafter, log stands for $\log_2$. The rootfinders in [26], [28] recursively combine *preprocessing* and *splitting*. Preprocessing algorithms compute the basic annulus A for balanced splitting with relative width

$$\psi = \frac{r^+}{r_-} \geq 1 + \frac{c}{n^d} \qquad (1.7)$$

for two real constants $c > 0$ and d. Splitting algorithms consist of computing a *crude initial splitting* and its *refinement* by nearly optimal Newton's iterative process. The refinement process has been fully developed in several papers (see [8], [12], [15], [17], [18], and [39]).

Let us first state the basic splitting results of Schönhage and Kirrinnis and then our main splitting theorem.

THEOREM 1.1. *[15], [39]. Given a polynomial p of (1.1), (1.5), a positive integer $k, k < n$, real $c > 0, d$,*

$$N = N(n, d) = \begin{cases} n & \text{for } d \leq 0, \\ n \log n & \text{for } d > 0, \end{cases} \qquad (1.8)$$

and $b \geq N$, an annulus $A = A(X, r_-, r^+)$ of (1.2), (1.7) such that

$$|z_j| \leq r_- \text{ for } j \leq k, \quad |z_j| \geq r^+ \text{ for } j > k, \qquad (1.9)$$

and two polynomials $\widetilde{F}$ (monic, of degree k, with all its zeros lying in the disc $D = D(X, r_-)$) and $\widetilde{G}$ (of degree $n-k$, with all its zeros lying outside the disc $D(X, r^+)$), satisfying

$$|p - \widetilde{F}\widetilde{G}| \leq 2^{-\tilde{c}N} |p| \qquad (1.10)$$

for a fixed and sufficiently large constant $\tilde{c}$, it is sufficient to perform $O((n \log n) \log b)$ ops with $O(b)$-bit precision or $O(\mu(bn))$ bit-operations for $\mu(b)$ of (1.4), to compute the coefficients of two polynomials $F^* = F^*(x)$ (monic, of degree k, and having all its zeros lying in the disc $D = D(X, r_-)$) and $G^* = G^*(x)$ (of degree $n - k$ and having all its zeros lying outside the disc D) such that

$$|p - F^* G^*| \leq 2^{-b} |p|. \qquad (1.11)$$

We use an equivalent version of the theorem where we relax assumption (1.5) and linearly transform the variable x (we shift X into the origin) to ensure that

$$X = 0, \quad q r_- = 1, \quad q = r^+, \quad \psi = q^2 \qquad (1.12)$$

for some $q > 1$. Then all concentric annuli $A(0, r_1, r_2)$ for $r_- \leq r_1 \leq r_2 \leq r^+$, including the unit circle $A(0, 1, 1) = C(0, 1) = \{x, |x| = 1\}$, split the polynomial p and separate the factors F^* and G^* from one another. The computation of the factors F^* and G^* of Theorem 1.1 satisfying (1.11) is called *splitting the polynomial p over the unit circle.*

THEOREM 1.2. *[39] (cf. [8], [17], [18], and Appendix A of [27]). Given a polynomial p of (1.1), a real N, and an annulus $A = A(0, r_-, r^+)$ such that (1.2), (1.7)–(1.9), (1.12) hold, it is sufficient to perform $O(M \log M)$ ops with $O(N)$-bit precision or $O((M \log M)\mu(N))$ bit-operations to compute the initial splitting polynomials $\widetilde{F}$ (monic, of degree k, with all zeros lying in the disc $D = D(0, 1)$) and $\widetilde{G}$ (of degree $n-k$, and with all zeros lying outside $D(0, 1)$) satisfying (1.10). Here, we have (cf. (1.7))*

$$M = n + N/(\psi - 1) = \begin{cases} O(n) & \text{for } d \leq 0, \\ O(n^{1+d} \log n) & \text{for } d > 0. \end{cases}$$

Thus the initial splitting can be computed in nearly optimal time (up to a polylog factor) if $d \leq 0$ but not if $d > 0$. Theorems 1.1 and 1.2 are implicit in [15] and [39] although the stated assumptions are slightly different and the op/precision count is not included. As in [26], [28], we rely on a lifting/descending process to reduce the case of positive d to the case of $d = 0$ but yield a substantially stronger result, that is, by the factor of n we improve the bounds on the computational precision and the Boolean cost versus [26], [28].

THEOREM 1.3. *Under the assumptions of Theorem 1.2, it is sufficient to perform $O((n \log n)(\log^2 n + \log b))$ ops with $O(b)$ precision or $O((\mu(b) n \log n)(\log^2 n + \log b))$ bit-operations to compute the coefficients of the two polynomials F^* and G^* of Theorem 1.1 satisfying (1.11)).*

Technically, in this part we focus on the refined analysis of the lifting/descending process, which, in spite of its crucial role in the design of nearly optimal polynomial rootfinders, remains essentially unknown to the computer algebra community. For instance, even in his most serious and comprehensive treatment of splitting a polynomial [15], P. Kirrinnis ignores the glaring flaw in the variation of this process presented in [24], although this process is a centerpiece of the paper [24], whose main result has been lost because of the flaw (see the appendix).

Our analysis (cf. also [33] or [5]) is technically involved but finally reveals surprising numerical stability (in terms of the asymptotic relative errors of the order 2^{-cn}) of Padé approximation (provided that the zeros of the input analytic function are isolated from its poles) and of Graeffe's lifting process, and this observation is a springboard for our current progress in polynomial factorization and rootfinding.

We refer the reader to [5] and [33] on further details.

Organization of Part I

In the next section we define our lifting/descending process, which splits a polynomial into two factors over a fixed zero-free annulus. We also estimate the arithmetic cost of the performance of this process and state the bound on the precision of its computation. The correctness of the algorithm under this precision bound is shown in section 3. The analysis includes the error estimates for the perturbation of the Padé approximation involved. In the appendix, we cover the extensions of our splitting over the unit circle to any basic circle (in part A) and to the complete numerical factorization of a polynomial (in part B) and we comment on related works (in parts C and D). The computation of the basic annuli for splitting is covered in Part II.

2. INITIAL SPLITTING VIA A LIFTING/ DESCENDING PROCESS

Theorem 1.2 enables splitting a polynomial p at a low cost over a circle well isolated from the zeros of p. Theorem 1.3 partly relaxes the isolation requirement. It is sufficient if the circle lies in the middle of a narrow zero-free annulus. Then we recursively apply the so called Graeffe's root–squaring process, whose each step squares the isolation ratio of the annulus. In $O(\log n)$ steps, the relative width of the root-free annulus grows from $1 + c/n^d$ to 4 and above, and we may apply Theorem 1.2, to split the lifted polynomial into two factors. Then we recursively descend down to the original polynomial by reversing Graeffe's lifting process. We observe that the input of every Graeffe step is defined by the $(n-k, k)$ entry of the Padé approximation table for a rational function formed by the output of this step. This immediately reduces every descending step to the computation of a Padé approximation. It is known that this computation has small arithmetic cost, but it is quite surprising that its asymptotic Boolean (bit-operation) cost turns out to be low as well. A detailed technical description of the algorithm supporting Theorem 1.3 follows next except that we omit the analysis of the Padé step (see [5]) or [33]), and then we estimate its arithmetic cost (in this section) and the Boolean cost (in section 3).

ALGORITHM 2.1. **Recursive lifting, splitting, and descending.**

INPUT: *positive c, r_-, r^+, real $\tilde{c}$ and d, and the coefficients of a polynomial p satisfying (1.1), (1.7), (1.9), and (1.12).*

OUTPUT: *polynomials F^* (monic and of degree k) and G^* (of degree $n-k$), split by the unit circle $C(0, 1)$ and satisfying bound (1.10) for $\epsilon = 2^{-\tilde{c}N}$ and N of (1.8).*

COMPUTATION:

Stage 1 (recursive lifting). Write $q_0 = p/p_n$, compute the integer

$$u = \lceil d \log n + \log(2/c) \rceil, \tag{2.1}$$

and apply u root-squaring Graeffe's steps

$$q_{l+1}(x) = (-1)^n q_l(-\sqrt{x}) q_l(\sqrt{x}), \quad l = 0, 1, \ldots, u-1. \tag{2.2}$$

(Note that $q_l = \prod_{i=1}^{n}(x - z_i^{2^l})$, $l = 0, 1, \ldots, u$, so $D(0, 1)$ is a ψ^{2^l}-isolated disc for q_l, for all l.)

Stage 2 (splitting q_u). Deduce from (2.1) that $\psi^{2^u} > 4$ and apply Theorem 1.2 to split the polynomial $p_u = q_u/|q_u|$ into two factors F_u^* and $\widetilde{G}_u$over the unit circle such that

$$|q_u - F_u^* G_u^*| = \epsilon_u |q_u|, \quad \epsilon_u \leq 2^{-CN} \tag{2.3}$$

for $G_u^* = |q_u| \widetilde{G}_u$ and a sufficiently large constant $C = C(c, d)$.

Stage 3 (recursive descending). Start with the latter splitting of q_u and recursively split the polynomials q_{u-j} of (2.2) over the unit circle, for $j = 1, \ldots, u$. Output the computed approximations $F^* = F_0^*$ and $G^* = p_n G_0^*$ to the two factors F and G of the polynomial $p = p_n q_0 = FG$. (The approximation error bounds are specified later on.)

Remark 2.2. *The algorithm applies Theorem 1.2 only at Stage 2, where the computations are not costly because the zeros of the polynomial p_u are isolated from the unit circle, due to (1.7), (1.9), and (1.12) for $1/(\psi - 1) = O(1)$ and p replaced by p_u.*

Let us specify Stage 3.

Stage 3 (recursive descending). Step j, $j = 1, 2, \ldots, u$. Stop where $j = u$; for $j < u$, go to the $(j+1)$-st step.

Input: the polynomial q_{u-j} (computed at Stage 1) and the computed approximations F^*_{u-j+1} and G^*_{u-j+1} to the factors F_{u-j+1} and G_{u-j+1} of the polynomial q_{u-j+1}, which is split over the unit circle. (The approximations are computed at Stage 2 for $j = 1$ and at the preceding, $(j-1)$-st, descending step of Stage 3 for $j > 1$.)

Computation: approximate the pair of polynomials $F_{u-j}(x)$ and $G_{u-j}(-x)$ as the pair filling the $(k, n-k)$-entry of the Padé approximation table for the meromorphic function. Given polynomials q_{u-j} and G^*_{u-j+1} (approximating the factor G_{u-j+1} of q_{u-j+1}), first approximate the polynomial $M_{u-j}(x) \bmod x^{n+1}$,

$$\begin{aligned} M_{u-j}(x) &= q_{u-j}(x)/G_{u-j+1}(x^2) \\ &= (-1)^{n-k} F_{u-j}(x)/G_{u-j}(-x). \end{aligned} \tag{2.4}$$

Then solve the Padé approximation problem (cf. Problem 5.2b (PADÉ) in [4, chapter 1]) or Problem 2.9.2 in [32] for the input polynomial $M_{u-j}(x) \bmod x^{n+1}$ to obtain the polynomials $F^*_{u-j} = F^*_{u-j}(x)$ (approximating F_{u-j}), $G^*_{u-j}(-x)$, and thus $G^*_{u-j} = G^*_{u-j}(x)$ (approximating G_{u-j}) such that

$$|F^*_{u-j} G^*_{u-j} - q_{u-j}| = \epsilon_{u-j} |q_{u-j}|, \quad \epsilon_{u-j} \le 2^{-\tilde{c}N}, \tag{2.5}$$

for $\tilde{c}$ of (1.10), where $q_{u-j} = F_{u-j} G_{u-j}$, $degF^*_{u-j} = k$, the polynomial F^*_{u-j} is monic, and $\deg G^*_{u-j} \le n - k$. Then improve the computed approximations of F_{u-j} by F^*_{u-j} and of G_{u-j} by G^*_{u-j}, by applying Theorem 1.1 with p replaced by q_{u-j}, F^* by F^*_{u-j}, and G^* by G^*_{u-j}. In the refinement, ϵ_{u-j} remains the value of the order of $1/2^{O(n \log n)}$ for $j < u$, whereas the bound $\epsilon_0 < 2^{-b}$ is ensured at the last (u-th) step.

Output of step j: Of the two computed factors, F^*_{u-j} and G^*_{u-j}, only the latter one is used at the subsequent descending step, though at the last step, both F^* and G^* are output.

The equations $G_{u-j+1}(x^2) = (-1)^{n-k} G_{u-j}(x) G_{u-j}(-x)$ and $\gcd(F_{u-j}(x), G_{u-j}(-x)) = 1$ together with (2.4) immediately imply the correctness of Algorithm 2.1 performed with infinite precision and no rounding errors provided that bound (2.5) holds true for $\epsilon_{u-j} = 0$ (that is, that $F^*_{u-j} = F_{u-j}$, $G^*_{u-j} = G_{u-j}$) for all j.

We next estimate the arithmetic complexity of Algorithm 2.1.

Stage 1: $O(un \log n) = O(n \log^2 n)$ ops at the $u = O(\log n)$ lifting steps, each is a polynomial multiplication (we use the FFT based algorithms).

Stage 2 (for $\epsilon_u = 1/2^{O(n \log n)}$): a total of $O(n \log^2 n)$ ops, by Theorems 1.1 and 1.2.

Stage 3: $O(n \log^2 n)$ ops for the computation of the polynomials $M_{u-j+1}(x) \bmod x^{n+1}$ for all j, $j = 1, \ldots, u$ (this is polynomial division modulo x^{n+1} for each j) and $O(n \log^3 n)$ ops for the computation of the $(k, n-k)$-th entries of the Padé approximation tables for the polynomials $M_{u-j+1}(x) \bmod x^{n+1}$ for $j = 1, \ldots, u$.

For every j, the latter computation is reduced to solving a nonsingular Toeplitz linear system of $n - k$ equations (see, e.g., [4, chapter 2], equation (5.6) for $z_0 = 1$ or Proposition 9.4 where $s(x) = 1$ or [32, Algorithm 2.11.1]); the Padé output entry is filled with a nondegenerating pair of polynomials $(F_{u-j}(x), G_{u-j}(-x))$. (Nonsingularity and nondegeneration follow because the polynomials $F_{u-j}(x)$ and $G_{u-j}(-x)$ have no common zeros and, therefore, have only constant common divisors; we extend this property to their approximations in the next section.) Moreover, the input coefficients of the auxiliary nonsingular Toeplitz linear systems (each of $n - k$ equations) are exactly the coefficients of the input polynomial $M_{u-j}(x) \bmod x^{n+1}$ of the Padé approximation problem.

To solve the u Toeplitz linear systems (where $u = O(\log n)$), we first symmetrize them and then apply the MBA algorithm of Morf and Bitmead/Anderson (cf. [4, chapter 2, Theorem 13.1] or [32, chapter 5]). The symmetrization ensures positive definiteness and therefore numerical stability [6]. $O(n \log^3 n)$ ops are sufficient in the u steps of Stage 3. Summarizing, we arrive at the *arithmetic cost estimates* of Theorem 1.3.

We perform all computations by Algorithm 2.1 with the precision of $O(n \log n)$ bits except for the refinement of the approximate initial splitting of the polynomial $q_0(x)$. There, we require (1.11) for a fixed $\epsilon = 2^{-b}$, $b \ge N$, and use computations with the b-bit precision. To prove Theorem 1.3, it remains to show that under the cited precision bounds, Algorithm 2.1 remains correct, that is , bound (2.5) holds for a fixed and sufficiently large $\tilde{c}$. We show this in the next section.

3. PRECISION AND COMPLEXITY ESTIMATES

Our goal is to prove that the computational precision of $O(N)$ bits and the bounds of order 2^{-cN} on the values ϵ_{u-j} of (2.5) for $j = 0, 1, \ldots, u$ are sufficient to support Algorithm 2.1. We first recall the following theorem.

Theorem 3.1. *[40, Theorem 2.7]. Let*

$$p = p_n \prod_{j=1}^{n} (x - z_j), \quad p^* = p^*_n \prod_{j=1}^{n} (x - z^*_j),$$

$$|p^* - p| \le \nu |p|, \quad \nu < 2^{-7n},$$

$$|z_j| \le 1, \ j = 1, \ldots, k; \quad |z_j| \ge 1, \ j = k+1, \ldots, n.$$

Then, up to reordering z^*_j, we have

$$|z^*_j - z_j| < 9 \sqrt[n]{\nu}, \ j = 1, \ldots, k;$$

$$|1/z_j^* - 1/z_j| < 9\sqrt[n]{\nu},\ j = k+1, \dots, n.$$

By applying the theorem for $p = q_{u-j} = F_{u-j}G_{u-j}$, $p^* = F_{u-j}^* G_{u-j}^*$, we obtain the following result.

Corollary 3.2. *Let (1.1), (1.9), (1.12), (2.1), and (2.5) hold and let $\epsilon_{u-j} < \min\{2^{-7n}, ((\psi-1)\theta/9)^n\}$ for all j and a fixed θ, $0 \le \theta < 1$. Then for all j, $j = 0, 1, \dots, u$, all zeros of the polynomials $F_{u-j}^*(x)$ and the reciprocals of all zeros of the polynomials $G_{u-j}^*(x)$ lie inside the disc $D(0, \theta + (1-\theta)/\psi)$. For $\psi - 1 \ge c/n^d$, $c > 0$, the latter properties of the zeros are ensured already where $\epsilon_{u-j} \le 1/n^{O(N)}$ for all j.*

Let us estimate the error of splitting $q_{u-j}(x)$ in terms of the approximation error for splitting $q_{u-j+1}(x)$.

Proposition 3.3. *Suppose that $|F_{u-j+1}^* G_{u-j+1}^* - q_{u-j+1}| \le \epsilon_{u-j+1}|q_{j-j+1}|$ for some real ϵ_{u-j+1} and a monic polynomial F_{u-j+1}^* of degree k. Let the Padé approximation problem be solved exactly (with the infinite precision and no rounding errors) for the input polynomial $M_{u-j}^*(x) =$ $(q_{u-j}(x)/G_{u-j+1}^*(x^2)) \bmod x^{n+1}$. Let F_{u-j}^*, G_{u-j}^* denote the solution polynomials and let ϵ_{u-j} be defined by (2.5). Then we have $|F_{u-j}^* - F_{u-j}| + |G_{u-j}^* - G_{u-j}| \le \epsilon_{u-j+1}2^{O(n \log n)}$, $\epsilon_{u-j} = \epsilon_{u-j+1}2^{O(n \log n)}$.*

Due to the latter proposition, it is sufficient to choose the value ϵ_{u-j} of (2.5) of the order of $\epsilon_{u-j+1}2^{-\tilde{c}N}$ for a large positive $\tilde{c}$ to ensure splitting q_{u-j} within an error bound (1.10), that is, small enough to allow the subsequent refinement based on Theorem 1.1.

The next theorem of independent interest is used in the proof of Proposition 3.3. It estimates the perturbation error of the Padé approximation problem. Generally, the input perturbation causes unbounded output errors but in our special case the zeros of the output pair of polynomials are isolated from the unit circle.

Theorem 3.4. *Let us be given two integers, k and n, $n > k > 0$, three positive constants C_0, γ, and ψ,*

$$\psi > 1, \tag{3.1}$$

and six polynomials F, f, G, g, M and m. Let the following relations hold

$$F = \prod_{i=1}^{k}(x - \hat{z}_i), \quad |\hat{z}_i| \le 1/\psi, \quad i = 1, \dots, k, \tag{3.2}$$

$$G = \prod_{i=k+1}^{n}(1 - x/\hat{z}_i), \quad |\hat{z}_i| \ge \psi, \quad i = k+1, \dots, n \tag{3.3}$$

(compare (1.9), (1.12)),

$$F = MG \bmod x^{n+1}, \tag{3.4}$$

$$F + f = (M+m)(G+g) \bmod x^{n+1}, \tag{3.5}$$

$$\deg f \le k, \tag{3.6}$$

$$\deg g \le n-k, \tag{3.7}$$

$$\begin{gathered} |m| \le \gamma^n(2 + 1/(\psi-1))^{-C_0 n}, \\ \gamma < \min\{1/128, (1-1/\psi)/9\}. \end{gathered} \tag{3.8}$$

Then there exist two positive constants C and C^* independent of n and such that if $|m| \le (2 + 1/(\psi-1))^{-Cn}$, then

$$|f| + |g| \le |m|(2 + 1/(\psi-1))^{C^* n}. \tag{3.9}$$

The proof of Theorem 3.4 is elementary but quite long. It can be found in [33] or [5] where the constant C_0 is specified.

Proof of Proposition 3.3. The relative error norms ϵ_{u-j} and ϵ_{u-j+1} are invariant in scaling the polynomials. For convenience, we drop all the subscripts of F, F^*, G, q and q^* and use scaling that makes the polynomials F, F^*, $G_{rev} = x^{n-k}G(1/x)$, and $G_{rev}^* = x^{n-k}G^*(1/x)$ monic, that is, $F = \prod_{j=1}^{k}(x - z_j)$, $F^* = \prod_{j=1}^{k}(x - z_j^*)$, $G = \prod_{j=k+1}^{n}(1 - x/z_j^*)$, $G^* = \prod_{j=k+1}^{n}(1 - x/z_j^*)$, $q = FG$, $q^* = F^*G^*$. The polynomials q and q^* are not assumed monic anymore (compare Remark 3.5.) Furthermore, by (3.1) – (3.3) and Corollary 3.2, we may assume that $|z_j| < 1$, $|z_j^*| < 1$, for $j \le k$, whereas $|z_j^*| > 1$, $|z_j| > 1$, for $j > k$. Therefore, $1 \le |F| < 2^k$, $1 \le |F^*| < 2^k$, $1 \le |G| < 2^{n-k}$, $1 \le |G^*| < 2^{n-k}$, $1 < |q| < 2^n$, $1 < |q^*| < 2^n$.

For any positive r, let us deduce that

$$\begin{aligned} &\left\| \frac{1}{G_{u-j+1}(x)} \bmod x^{r+1} \right\| \le \left\| (1-x)^{k-n} \bmod x^{r+1} \right\| \\ &= \sum_{i=0}^{r} \binom{n-k+i-1}{n-k-1} = \binom{n-k+r}{r} < 2^{n-k+r}, \end{aligned} \tag{3.10}$$

Indeed, write $(-x)^{n-k}/G_{n-k}(x) = \sum_{i=0}^{\infty} g_i/x^i$. Observe for each i that $|g_i|$ reaches its maximum where $z_i = 1$, that is, where $(-x)^{n-k}/G_{n-k}(x) = x^{n-k}/(1-x)^{n-k}$, and (3.10) follows.

Likewise, we have

$$\|(1/G_{u-j+1}^*(x)) \bmod x^r\| < 2^{n-k+r}.$$

We apply a bound of section 10 of [39] to obtain that

$$|G_{u-j+1}^* - G_{u-j+1}| \le \epsilon_{u-j+1}2^{O(N)}.$$

Now write

$$\Delta_{u-j+1} = \left(\frac{1}{G_{u-j+1}^*} - \frac{1}{G_{u-j+1}} \right) = \frac{G_{u-j+1} - G_{u-j+1}^*}{G_{u-j+1}G_{u-j+1}^*},$$

summarize the above estimates, and obtain that

$$\|\Delta_{u-j+1}(x) \bmod x^r\| \le \epsilon_{u-j+1}2^{O(n \log n)}$$

for $r = O(n)$.

Next write $m_{u-j} = m_{u-j}(x) = (M_{u-j}^*(x) - M_{u-j}(x)) \bmod x^{n+1}$ and combine our latter bound with (2.4) and with the bound $|q_{u-j}| \le 2^n$ to obtain that $|m_{u-j}| \le \epsilon_{u-j+1}2^{O(N)}$. By combining this estimate with the ones of Theorem 3.4, we obtain the first bound of Proposition 3.3,

$$\Delta_{F,G} = |F_{u-j}^* - F_{u-j}| + |G_{u-j}^* - G_{u-j}| \le \epsilon_{u-j+1}2^{O(N)}.$$

Now we easily deduce the second bound,

$$\begin{aligned}\epsilon_{u-j} &= |F^*_{u-j}G^*_{u_j} - F_{u-j}G_{u-j}| \\ &\leq |F^*_{u-j}(G^*_{u-j} - G_{u-j}) + (F^*_{u-j} - F_{u-j})G_{u-j}| \\ &\leq |F^*_{u-j}| \cdot |G^*_{u-j} - G_{u-j}| + |F^*_{u-j} - F_{u-j}| \cdot |G_{u-j}| \\ &\leq \max\left\{|F^*_{u-j}|, |G_{u-j}|\right\} \Delta_{F,G} \\ &\leq \epsilon_{u-j+1} 2^{O(N)}. \quad \square\end{aligned}$$

Similarly to Proposition 3.3, we may prove that any perturbation of the coefficients of the polynomial q_{u-j} within the relative norm bound of the order $1/2^{O(N)}$ causes a perturbation of the factors of q_{u-j} within the relative error norm of the same order.

Proposition 3.3 and Theorem 3.4 together show that the relative errors of the order of $O(N)$ bits do not propagate in the descending process of Stage 3 of Algorithm 2.1. We proved Theorem 3.4 in [33] (and included the proof in [5]).

To complete the proof of Theorem 1.3, it remains to show that the relative precision of $O(N)$ bits for the output of the descending process of Algorithm 2.1 can be supported by the computations with rounding to the precision of $O(N)$ bits. To yield this goal, one may apply the tedious techniques in [38] (cf. also [15] and [39]). Alternatively we apply the backward error analysis to all the polynomial multiplications and divisions involved, to simulate the effect of rounding errors of these operations by the input perturbation errors. This leads us to the desired estimates simply via the invocation of Theorem 3.4 and Proposition 3.3, except that we need some distinct techniques at the stages of the solution of Toeplitz linear systems of equations associated with the Padé problem.

To extend our analysis to these linear systems, we recall that they are nonsingular because the Padé problem does not degenerate in our case. Moreover, Theorem 3.4 bounds the condition number of the problem. Furthermore, we solve the Padé problem by applying the cited MBA algorithm to the symmetrized linear systems. (The symmetrization squares the condition number, which requires doubling the precision of the computation, but this is not substantial for proving our estimate of $O(N)$ bits.) We then recall that the algorithm only operates with some displacement generators defined by the entries of the Padé input, $M^*_{u-j}(x) \bmod x^{n+1}$, and is proved to be numerically stable [6]. It follows that $O(N)$-bit precision of the computation is sufficient at the stages of solving Padé problems too, and we arrive at Theorem 1.3. $\square$

Remark 3.5. *One could have expected even a greater increase of the precision required at the lifting steps of (2.2). Indeed, such steps generally cause rapid growth of the ratio of the absolutely largest and the absolutely smallest coefficients of the input polynomial. Such a growth, however, does not affect the precision of computing because all our error norm bounds are relative to the norms of the polynomials. Technically, to control the output errors, we apply scaling, to make the polynomials F, F^*, G_{rev} and G^*_{rev} monic, and then continue as in the proof of Proposition 3.3, where the properties (1.9) of the zeros of the input polynomials are extended to the approximations to the zeros, due to Corollary 3.2.*

Part II: Computing a Basic Annulus for Splitting

1. THE MAIN RESULTS AND THE ORGANIZATION OF PART II

The algorithms of Part I enable us to reduce the approximation of the zeros of a polynomial

$$p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{j=1}^{n} (x - z_j), \quad p_n \neq 0, \tag{1.1}$$

to the computation of a basic annulus over which the polynomial $p(x)$ can be split effectively into the product of two nonlinear factors, $F(x)$ and $G(x)$. In this part of the paper, we improve the algorithm of [26], [28], which computes a basic annulus and, moreover, ensures that the resulting splitting is *a-balanced*, that is,

$$(1-a)n/2 \leq \deg F(x) \leq (1+a)n/2, \tag{1.2}$$

where a is any fixed constant from the interval

$$5/6 \leq a < 1. \tag{1.3}$$

We use the definitions of Part I, including the concepts of *ops* (that is, arithmetic operations + comparisions + the computation of $|z|$ or $|z|^{1/k}$ for complex numbers z and integers $k > 1$), *the splitting of polynomials over (zero-free) annuli, the relative width* $\rho(A) \geq 1$ of an annulus A (the ratio of the radii of its two boundary circles), ψ*-isolated discs* (the internal discs of zero-free annuli having a relative width ψ), and the *norm* $\|\sum_i u_i x^i\| = \sum_i |u_i|$. log still means $\log_2$.

Under the above assumptions, in each step of recursive splitting, we compute two factors of the input polynomial whose degrees are at most $(1+a)n/2$ (for instance, $11n/12$ for $a = 5/6$). Then we apply the estimates from Part I for the computational complexity of splitting a polynomial over a fixed circle. This gives us upper bounds on the overall arithmetic and Boolean computational cost of the complete factorization of the polynomial $p(x)$ into the product of linear factors and of the approximation of well- and ill-conditioned polynomial zeros. All bounds are optimal up to polylogarithmic factors.

Theorem 1.1. *Let* $p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{j=1}^{n} (x - z_j)$, $p_n \neq 0$, *be a polynomial of (1.1) of degree n given with its coefficients. Let*

$$|z_j| \leq 1 \text{ for all } j. \tag{1.4}$$

Let b be a fixed real number, $b \geq n \log n$. *Then complex numbers* z^*_j, $j = 1, \dots, n$, *satisfying*

$$\|p(x) - p_n \prod_{j=1}^{n} (x - z^*_j)\| \leq 2^{-b} \|p(x)\| \tag{1.5}$$

can be computed by using $O((n \log^2 n)(\log^2 n + \log b))$ *ops performed with the precision of* $O(b)$ *bits or by using* $O((n \log^2 n)(\log^2 n + \log b)\mu(b))$ *bit-operations for* $\mu(b)$ *denoting the bit-operation cost of performing a single op with the b-bit precision,*

$$\mu(b) = O((b \log b) \log \log b). \tag{1.6}$$

By a theorem from [39, section 19], the approximate factorization (1.5) defines approximations z_j^* to the zeros z_j of $p(x)$ satisfying

$$|z_j^* - z_j| < 2^{2-b/n}, \quad j = 1, \dots, n. \tag{1.7}$$

Corollary 1.2. *Under the assumptions of Theorem 1.1, its cost bounds apply to the task of computing approximations z_j^* to all the zeros z_j of a polynomial $p(x)$, where the approximation errors are bounded according to (1.7).*

Bound (1.7) covers the worst case polynomials $p(x)$ whose zeros may be ill-conditioned, that is, form clusters. The recovery of well-conditioned (isolated) zeros of $p(x)$ from factorization (1.5) has approximation error of the order of 2^{-b}.

With no preliminary knowledge about how well (or poorly) the zeros of a given polynomial $p(x)$ are isolated from each other, one may apply the algorithm supporting Theorem 1.1 and Corollary 1.2 and get the isolation information by examining the discs $D(z_j^*, 2^{2-b/n})$. To refine the bounds of (1.7), one may apply, for instance, the root radii algorithms in [39] and/or some modifications of the Weierstrass method (cf. [5]).

The estimates of Theorem 1.1 and Corollary 1.2 are nearly optimal. Indeed, even the approximation of a single zero of a polynomial $p(x)$ requires at least $(n+1)/2$ arithmetic operations. This follows because the approximation involves $n+1$ coefficients of $p(x)$, whereas each arithmetic operation has two operands and, therefore, may involve at most two parameters. To approximate the n zeros, we need at least n arithmetic operations because the algorithm must output n values that are generally distinct. Therefore, the arithmetic cost bound of Theorem 1.1 is optimal up to polylogarithmic factors in n. So is also the Boolean cost bound, due to Fact 1.1 in [28]. We also recall the lower bound $\Omega(\log b)$ in [37], on the number of arithmetic operations required for the approximation of even a single zero of $p(x)$ within 2^{-b} under the normalization assumption that $|z_i| \le 1$ for all i.

Remark 1.3. *We deduce our bit-operation (Boolean) cost bounds simply by combining the ops and precision bounds and the estimate (1.6) on the bit-operation (Boolean) cost of performing an op with the b-bit precision. (To apply the latter estimate, known for the bit-operation cost of an arithmetic operation with integers performed modulo $2^b + 1$, truncate real and complex operands to b bits and then scale them.) This approach can be immediately extended to yield bit-operation (Boolean) cost estimates based on other known upper bounds on $\mu(b)$ (cf. [5]). It seems that a small further decrease (by the factor of $O(\log n)$) of our bit-operation cost estimates is possible if one applies the refined integer arithmetic based on the binary segmentation techniques (cf. [4, section 3.3], [15], [38], [39], [15]).*

We organize Part II as follows. In the next section, we define the basic concept of balanced splitting annuli and discs. In section 3, we compute basic splitting annuli for a large class of input polynomials; for the remaining polynomials our algorithms confine most of their zeros to a small disc. In sections 4 and 5, we recall the results in [23] on the computation of the zeros of higher order derivatives $p^{(l)}(x)$ of an input polynomial $p(x)$ as a means of balancing the degrees in splitting. In section 6, we yield the same goal without computing the zeros of $p^{(l)}(x)$, which enables us to decrease the computational cost dramatically and to arrive at Theorem 1.1.

2. SOME BASIC DEFINITIONS AND RESULTS

To reach the (nearly optimal) estimates of Theorem 1.1, one must balance the degrees of the two output factors in each step of the recursive splitting. If, on the contrary, each splitting produces a linear factor, then $n-1$ splittings and at least the order of n^2 arithmetic operations are necessary.

It can be very hard to ensure balanced splitting, however. For example, for a polynomial $p(x) = \prod_{i=1}^{k}(x - 2^{-i^3} - 5/7)G(x)$, where $G(x)$ is a polynomial of degree $n-k = n^{1/3}$, one must separate from each other some zeros of $p(x)$ lying in the same disc of radius $1/2^{cn^3}$, for a fixed positive c. (By following [28], we will say that $p(x)$ has a *massive cluster* of zeros in such cases.) Then, to yield the balanced splitting, one must perform computations with a precision of the order of n^4 bits, even if we are only required to approximate the zeros of $p(x)$ within the error tolerance 2^{-10n}. Such a high precision of computing would not allow us to reach the Boolean complexity bounds of Theorem 1.1.

We salvage the optimality (up to polylog factors) only because we do not compute balanced splitting in this case. Indeed, the same point $z = 5/7$ approximates (within 2^{-10n}) all but $n - l = O(n^{1/3})$ zeros of $p(x)$, and it remains to approximate the remaining $n - l = O(n^{1/3})$ zeros of $p(x)$ by working with a polynomial of a degree $O(n^{1/3})$, obtained as the quotient of numerical division of $p(x)$ by $(x - 5/7)^l$.

Generalizing the latter recipe, we detect massive clusters and approximate their zeros without computing balanced splitting of a given polynomial. Formally, we introduce the concepts of (a, ψ)*-splitting annuli* (basic for balanced splittings) and (a, B, ψ)*-splitting discs* (each covering a massive cluster of the zeros to be approximated by a single point, without computing a balanced splitting).

Definition 2.1. *A disc $D(X, \rho) = \{x, |x - X| \le \rho\}$ is called an (a, B, ψ)*-splitting disc *for a polynomial $p(x)$ if it is both ψ-isolated and contains more than $(3a-2)n$ zeros of $p(x)$ and if ρ satisfies the relations*

$$\rho \le 2^{-B}. \tag{2.1}$$

An annulus $A(X, \rho_-, \rho_+) = \{x, \ \rho_- \le |x - X| \le \rho_+\}$ is called an (a, ψ)-splitting annulus *for $p(x)$ if it is free of the zeros of $p(x)$ and if its internal disc $D(X, \rho)$ contains exactly k zeros of $p(x)$ (counted with their multiplicities) where $\rho_+ \ge \psi\rho_-$ and*

$$(1-a)n/2 \le k \le (1+a)n/2 \tag{2.2}$$

(compare (1.1)). In the latter case we also call the disc $D(X, \rho_-)$ an (a, ψ)-splitting disc *for the polynomial $p(x)$.*

A disc containing exactly k zeros of $p(x)$ for k satisfying bounds (2.2) is called a-balanced.

DEFINITION 2.2. *The j-th root radius for $p(x)$ is the distance r_{n+1-j} from the origin to the j-th closest root (zero) of $p(x)$. (We have $r_{n+1-j} = |z_j|$, $j = 1, \dots, n$, if the zeros z_j of $p(x)$ are enumerated so that $|z_1| \geq |z_2| \geq \dots \geq |z_n|$.) We write $r_0 = \infty$, $r_{n+1} = 0$, and $r_j(X)$ for the j-th root radius of the polynomial $q(x) = p(x + X)$, obtained from $p(x)$ when the origin is shifted to a complex point X.*

We use the following auxiliary results.

PROPOSITION 2.3. *[39] (cf. also [31]). $O(n \log^2 n)$ ops performed with $O(n)$-bit precision are sufficient to approximate within relative error bound c/n^d (for any fixed pair of $c > 0$ and $d \geq 0$) all root radii r_j of a polynomial $p(x)$, $j = 1, \dots, n$, as well as all root radii $r_j(X)$ of $q(x) = p(x + X)$ for $j = 1, \dots, n$ and any fixed complex X.*

THEOREM 2.4. *[33]. Suppose that we are given a polynomial $p(x)$ of (1.1), the real constants b, c and $d, b > 1, c > 0$, and a splitting annulus $A(0, \psi, 1/\psi)$ where*

$$\psi \geq 1 + c/n^d. \tag{2.3}$$

Then it is sufficient to perform $O((n \log n)(\log^2 n + \log b))$ ops with $O(b)$ precision (which amount to $O(\mu(b) n \log n)(\log^2 n + \log b))$ bit-operations, for $\mu(b)$ of (1.6)) to compute two polynomials $F(x)$ and $G(x)$ with all their zeros lying in the disc $D(0, 1)$ and outside it, respectively, and such that

$$deg(F(x)G(x)) = n, \quad \|F(x)G(x) - p(x)\| \leq 2^{-b} \|p(x)\|.$$

REMARK 2.5. *We apply a simple extension of this theorem where the annulus $A(0, \psi, 1/\psi)$ is replaced by any other splitting annulus with a relative width ψ satisfying (2.3). The extension is based on the respective linear transformations of the variable x (cf. Appendix A).*

3. BASIC SPLITTING ANNULI OR LARGE ROOT CLUSTERS

Let us fix a positive a of (1.3) and let us write

$$\psi = 1 + c/n, \tag{3.1}$$

for a fixed small positive constant c (to be estimated later on). For a large class of polynomials $p(x)$, their (a, ψ)-splitting discs (and, consequently, their balanced splitting) can be computed immediately by means of the approximation of root radii. Indeed, apply Proposition 2.3 to compute an (a, ψ)-splitting disc for $p(x)$. Write

$$g(a) = \lfloor (1 - a)n/2 \rfloor, \quad h(a) = g(a) + \lfloor an \rfloor, \tag{3.2}$$

so $0 \leq (1 + a)n/2 - h(a) < 2$,

$$g(a) \geq \lfloor n/12 \rfloor, \quad h(a) \geq \lfloor n/12 \rfloor + \lfloor 5n/6 \rfloor \text{ for } a \geq 5/6,$$

and let r_i^+ and r_i^- denote the computed upper and lower estimates for $r_i = |z_i|$, $i = 1, \dots, n$. We require that

$$r_i^+ / r_i^- \leq \psi^* = 1 + (c/n)^2, \quad i = 1, \dots, n, \tag{3.3}$$

for the same fixed positive c, and we observe that the discs $D(0, r_i^+)$ are (r_{i-1}^- / r_i^+)-isolated, for all i. If

$$r_{i-1}^- / r_i^+ \geq \psi, \tag{3.4}$$

for at least one choice of an integer i satisfying

$$g(a) \leq n + 1 - i < h(a), \tag{3.5}$$

then the disc $D(0, r_i^+)$ is both a-balanced, due to (3.2), and ψ-isolated, due to (3.4).

This approach yields the desired (a, ψ)-splitting discs for a very large class of the input polynomials $p(x)$, that is, for those for which bound (3.4) holds for some integer i satisfying (3.5). To yield *Universal Rootfinders* for all input polynomials $p(x)$, it remains to treat the opposite case where bound (3.4) holds for none i of (3.5). In this case, we still make some progress based on Proposition 2.3. Indeed, observe that at least $h(a) - g(a) + 1 = \lfloor an \rfloor + 1$ zeros of $p(x)$ lie in the closed annulus,

$$A = \{x : r_{n+1-g(a)}^- \leq |x| \leq r_{n+1-h(a)}^+\}, \tag{3.6}$$

and recall that we have the bounds $r_{i-1}^- < \psi r_i^+$ for all i of (3.5). Together with (3.3), these bounds imply that the relative width of the annulus A satisfies the bound

$$r_{n+1-h(a)}^+ / r_{n+1-g(a)}^- \leq (\psi \psi^*)^{h(a) - g(a) + 1}. \tag{3.7}$$

Now we apply Proposition 2.3 twice, for the origin shifted into the points $2r_{n+1-h(a)}^-$ and $2r_{n+1-h(a)}^- \sqrt{-1}$. Then we either compute a desired (a, ψ)-splitting disc for $p(x)$ or arrive at two additional narrow annuli of radii at most $3r_{n+1-h(a)}^-$, each having a relative width of at most $(\psi \psi^*)^{h(a) - g(a) + 1}$ and each containing at least na zeros of $p(x)$. Our current goal is the determination of an (a, ψ)-splitting disc for $p(x)$, so it is sufficient to examine the latter case, where each of the three narrow annuli contains more than na zeros of $p(x)$.

We have the following simple but useful result, which we only need for $h = 3$.

PROPOSITION 3.1. *[23]. Let $S_1, S_2, \dots, S_h$ denote h finite sets. Let U denote their union and I their intersection. Then*

$$|I| \geq \sum_{i=1}^{h} |S_i| - (h - 1)|U|,$$

where $|S|$ denotes the cardinality of a set S.

Due to Proposition 3.1, the intersection of the three narrow annnuli contains more than $(3a - 2)n \geq n/2$ zeros of $p(x)$ (compare (1.4)). Since the annuli are narrow, we include their intersection into a sufficiently small *covering disc*, $D = D(Y, r)$. We ensure that $r < 0.1 r_{n+1-h(a)}^-$ (say), by choosing the constant c in (3.1) and (3.3) sufficiently small.

We could have decreased ψ to $1 + c/n^d$ for a small positive c and $d > 1$ and consequently decreased the radius of the disc to $O(r_{n+1-h(a)}^- / n^{d-1})$, but then an extension of Theorem 1.1 would be required to avoid a dramatic growth of the computational cost estimates in the subsequent construction and to confine the growth to the extra factor of $\log n$ (see Remark 6.2). Thus we stay with ψ of (3.1) but shift the origin into the center Y of the disc D and apply the same construction again. Furthermore, we repeat this process recursively until we obtain either a desired (a, ψ)-splitting disc for $p(x)$ or a (small) covering disc that contains more than

$(3a-2)n$ zeros of $p(x)$ and has a radius r bounded from above by $r^-_{n+1-h(a)}(0)/n^d$ for a fixed positive d. This radius may be small enough to enable the computation of an (a,B,ψ)-splitting disc. We always check if this is the case for each computed radius r (see Remark 3.5), but generally we cannot count on such a rapid success. Hereafter, we refer to this recursive computation as ALGORITHM 3.2.

PROPOSITION 3.3. *Write $X_0 = 0, r_0 = r^+_{n+1-h(a)}$ and let $D(X_i, r_i)$ denote the output covering disc of the i-th recursive step of Algorithm 3.2. Then we have $5r_i \le r_{i-1}$ and $|X_i| \ge r_0/2$ for $i = 1, 2, \dots$ provided that the constant c of (3.1) and (3.3) has been chosen small enough.*

PROOF. Let w_{i-1} denote the width of the narrow annulus A_i centered in X_i and computed at the $(i-1)$-st recursive step. By the construction of this annulus, we have

$$w_{i-1} \le (\psi\psi^*)^{h(a)-g(a)+1} - 1)r_{i-1}$$

where $h(a) - g(a) = \lfloor an \rfloor$ (cf. (3.2)). Clearly, $(\psi\psi^*)^{an} \to 1$ as $c \to 0$ for c of (3.1) and (3.3). Therefore, $w_{i-1}/r_{i-1} \to 0$ as $c \to 0$, that is, we may assume that $w_{i-1} \le \nu^2 r_{i-1}$ for any fixed positive ν. It is easy to verify that the intersection of the annulus A_i with the two other annuli computed at the same recursive step of Algorithm 3.2 must have diameter at most $\mu\sqrt{w_{i-1}r_{i-1}}$ for some fixed constant μ. Therefore, the radius r_i of the output covering disc (covering this intersection) is less than $\mu\nu r_{i-1}$. It remains to choose $\nu < 0.2/\mu$ to obtain that $5r_i \le r_{i-1}$. On the other hand, we have $|X_i - X_{i-1}| \to r_{i-1}$ as $c \to 0$, for $i = 1, 2, \dots$. Together with the bound $5r_i \le r_{i-1}$ and equation $X_0 = 0$, this implies that $|X_i| \ge r_0/2$ for $i = 1, 2, \dots$. □

COROLLARY 3.4. *Under the assumptions of Proposition 3.3, we have $2|X_i| \ge 5^i r_i$, for $i = 1, 2, \dots$.*

For a large class of input polynomials $p(x)$, Algorithm 3.2 outputs (a,ψ)-splitting discs, thus completing our task. In the remaining case, a covering disc D of smaller size is output. We may use the center of the disc D as a generally crude approximation to more than $n/2$ zeros of $p(x)$. The same algorithm can be extended to improve the latter approximations, decreasing the approximation errors with linear rate. This is too slow for us, however. We follow a distinct strategy: we specify and satisfy a condition under which Algorithm 3.2 never outputs a disc that covers the intersection of the three narrow annuli, so a desired (a,ψ)-splitting disc must be output.

REMARK 3.5. *Application of the root radii algorithm enables us to compute a desired (a,B,ψ)-splitting disc for $p(x)$ (see Definition 2.1) as soon as we detect that the value $B_k = 2^{-B}/(\psi\psi^*)^{n-k+1}$ exceeds the radius r of some computed disc $D(X,r)$ containing k zeros of $p(x)$ where $k > (3a-2)n$. Indeed, in this case, we shift the origin into the point X, compute the values r^-_i and r^+_i for $i = 1, 2, \dots, n-k+1$ and write $r^-_0 = \infty$. Then we choose the maximal i such that $i \le n-k+1$ and $r^-_{i-1}/r^+_i \ge \psi$ and observe that $r^+_i \le 2^{-B}$ and the disc $D(X, r^+_i)$ for such i is ψ-isolated and, therefore, is a desired (a,B,ψ)-splitting disc for $p(x)$. The comparison of the above values B_k with the radii of all computed discs containing more than $k \ge (3a-2)n$ zeros of $p(x)$ is assumed by default to be a part of all our algorithms (to simplify their description, we do not cite this comparison explicitly). Without making these comparisons, we would have lost our control over the precision and the Boolean cost of computing and would have allowed them to blow-up.*

4. A (T,S)-CENTER OF A POLYNOMIAL AS THE ZERO OF ITS HIGHER ORDER DERIVATIVE

We recall the following result from [7].

THEOREM 4.1. *For any integer l satisfying $0 < l < n$, for every disc $D(X,r)$ containing at least $l+1$ zeros of a polynomial $p(x)$ of degree n, and for any $s \ge 3$ if $l = n-1$ and any $s \ge 2 + 1/\sin(\pi/(n-l))$ if $l < n-1$, the disc $D(X,(s-2)r)$ contains a zero of $p^{(l)}(x)$, the l-th order derivative of $p(x)$.*

REMARK 4.2. *Theorem 4.1 extends to the complex polynomials Rolle's classical theorem about a zero of the derivative of a real function. A distinct and much earlier extension of this theorem to the complex case, due to A. Gel'fond [11], supports our nearly optimal asymptotic complexity estimates of Theorem 1.1 as well, although with slightly larger overhead constants hidden in the "O" notation of these estimates versus the case where we rely on Theorem 4.1. On the other hand, application of more advanced techniques in [7] enables a further decrease of the parameter s of Theorem 4.1 and, consequently, a further decrease of the latter constants. Namely, by using nontrivial techniques based on properties of symmetric polynomials, the result of Theorem 4.1 was extended in [7] to any s exceeding $2 + c\max\{(n-l)^{1/2}(l+1)^{-1/4}, (n-l)(l+1)^{-2/3}\}$, for $l = 2, 3, \dots, n-1$ and for some constant c. This extension allows one to replace s of Theorem 4.1, of the order of n, by s of the order of $n^{1/3}$.*

Hereafter, we assume that

$$l = \lfloor (3a-2)n \rfloor, \quad n - l = \lceil (3-3a)n \rceil, \tag{4.1}$$

and s satisfies the assumption of Theorem 4.1. By combining (1.3) and (4.1), we obtain that $l \ge \lfloor n/2 \rfloor$, $l + 1 > n/2$. In particular, one may choose

$$a = 5/6, \quad l = \lfloor n/2 \rfloor, \quad n - l = \lceil n/2 \rceil. \tag{4.2}$$

DEFINITION 4.3. *[23]. A disc $D(X,r)$ is called* t-full *if it contains more than t zeros of $p(x)$. A point Z is called a* (t,s)-center for $p(x)$ *if it lies in the dilation $D(X,sr)$ of any t-full disc $D(X,r)$.*

PROPOSITION 4.4. *[23]. Let $t \ge n/2$ and let $s > 2$. If a complex set S has a nonempty intersection with the dilation $D(X,(s-2)r)$ of any t-full disc $D(X,r)$, then such a set S contains a (t,s)-center for $p(x)$.*

PROOF. Let $D(X,r)$ be a t-full disc for $p(x)$ of the minimum radius and let Z be a point of the set S lying in the disc $D = D(X,(s-2)r)$. Let $D(Y,R)$ be another t-full disc for $p(x)$. Then $R \ge r$, and since $t \ge n/2$, this disc intersects $D(X,r)$. Therefore, the disc $D(Y,sR)$ covers the disc D and, consequently, the point Z, which is, therefore, a (t,s)-center for $p(x)$. □

Proposition 4.4 and Theorem 4.1 together imply the next result.

Corollary 4.5. *If s satisfies the assumptions of Theorem 4.1 for $n+1 > l+1 > n/2$, then at least one of the $n-l$ zeros of the l-th order derivative of $p(x)$ is an (l,s)-center for $p(x)$.*

5. (T,S)-CENTERS AND SPLITTING A POLYNOMIAL

Now suppose that we apply the recursive algorithm of section 3 in the case where the origin is initially shifted into a (t,s)-center Z for $p(x)$ and where $t/n = 3a-2 \geq 1/2$. Then after sufficiently many recursive steps, an (a,ψ)-splitting disc must be output. Indeed, otherwise, according to Corollary 3.4, for every positive i the i-th recursive step must output a covering disc $D(X_i, r_i)$ containing more than $(3a-2)n$ zeros of $p(x)$ where $5^i r_i \leq 2|X_i|$. Then it follows that

$$sr_i < |X_i|, \tag{5.1}$$

already for some $i = O(\log s)$. The latter inequality implies that the origin cannot lie in the disc $D(X_i, sr_i)$, in contradiction to our assumption that the origin is (or has been shifted into) a (t,s)-center for $p(x)$.

This gives us an algorithm (hereafter referred to as Algorithm 5.1) that computes an (a,ψ)- or an (a,B,ψ)-splitting disc for $p(x)$ as soon as we have a (t,s)-center for $p(x)$ where $t \geq n/2$.

It is easy to extend Algorithm 5.1 to the case where an approximation to a (t,s)-center for $p(x)$ is available within a small absolute error, say, being less than

$$\rho^* = 2^{-2B}/s. \tag{5.2}$$

The extension relies on the following result.

Proposition 5.2. *Suppose that an unknown $((3a-2)n,s)$-center for $p(x)$ lies in a disc $D(0,\rho^*)$. Suppose that Algorithm 5.1 applied at the origin (rather than at such a center) does not output an (a,ψ)-splitting disc for $p(x)$ but yields a covering disc $D = D(X,r)$, which is $((3a-2)n)$-full for $p(x)$. Then*

$$|X| \leq sr + \rho^*. \tag{5.3}$$

Proof. A $((3a-2)n,s)$-center for $p(x)$ lies in both discs $D(X,sr)$ and $D(0,\rho^*)$. These two discs have a nonempty intersection because $3a-2 \geq 1/2$, and hence, $|X| \leq sr + \rho^*$. □

By Proposition 5.2, application of Algorithm 5.1 should output a desired (a,ψ)- or (a,B,ψ)-splitting disc for $p(x)$ as soon as we have $sr_i < |X_i| - \rho^*$, which for a small ρ^* of (5.2) is almost as mild a bound as (5.1).

Due to Corollary 4.5, an (l,s)-center for $p(x)$ can be found among the $n-l$ zeros of the l-th order derivative $p^{(l)}(x)$ for l of (4.1). Suppose that the set, Z_l^*, of sufficiently close approximations to these zeros within ρ^* of (5.2) is available, but we do not know which of them is a (t,s)-center for $p(x)$, for $t \geq n/2$. Then, clearly, we still may compute a desired splitting disc by applying Algorithm 5.1 with the origin shifted into each of the $n-l$ approximations to the $n-l$ zeros of $p^{(l)}(x)$. Alternatively, we may apply an implicit binary search [23], which enables us to shift the origin into at most $\lceil \log(n-l) \rceil$ candidate approximation points Y_i. We call the latter algorithm (using binary search) Algorithm 5.3.

In spite of the acceleration by roughly the factor of $(n-l)/\log(n-l)$ versus application of Algorithm 5.1 at every point of S_0, the latter algorithm still reduces the approximation of the zeros of $p(x)$ to the approximation of the zeros of a higher order derivative $p^{(l)}(x)$ (at first) and then of two factors of $p(x)$, $F(x)$ and $G(x)$. Due to the extra stage of the approximation of the zeros of $p^{(l)}(x)$, which precedes the computation of a splitting disc for $p(x)$, the overall upper bounds on both sequential and parallel time of polynomial rootfinding increase by the factor of n^δ for some positive δ. In the next section, we show how to avoid this costly stage.

6. THE ROOTS OF HIGHER ORDER DERIVATIVES ARE NOT REQUIRED

Suppose that we have an (a,ψ)-splitting disc, $D(Y,R)$, for the l-th order derivative $p^{(l)}(x)$ for l of (4.1). Then we may shift the origin into Y and apply Algorithm 3.2, repeating the recursive process until either a desired (a,ψ)-splitting disc for $p(x)$ is computed or the dilation $D(X_i, sr_i)$ of a covering disc $D(X_i, r_i)$ lies either entirely in the disc $D(Y,\psi R)$ or entirely in the exterior of the disc $D(Y,R)$. The latter property of the dilation of the disc $D(X_i,r_i)$ is clearly ensured if the width $(\psi-1)R$ of the computed annulus $\{x : R \leq |x| \leq \psi R\}$ (which is free of the zeros of $p^{(l)}(x)$) exceeds the diameter $2sr_i$ of the disc $D(X_i, sr_i)$.

Let z denote a (t,s)-center for $p(x)$ such that $p^{(l)}(z) = 0$, $t \geq n/2$. Let $p^{(l)}(x) = f_l(x)g_l(x)$, where $f_l(x)$ and $g_l(x)$ are two polynomials, $f_l(x)$ has all its zeros in the disc $D(Y,R)$, and $g_l(x)$ has no zeros in the disc $D(Y,\psi R)$. Then we have $f_l(z) \neq 0 = g_l(z)$ if the dilation $D(X_i, sr_i)$ of the covering disc $D(X_i,r_i)$ has only empty intersection with the disc $D(Y,R)$, and we have $f_l(z) = 0 \neq g_l(z)$ if $D(X_i,sr_i) \subseteq D(Y,\psi R)$. Therefore, the considered application of Algorithm 3.2 enables us to discard one of the two factors, $f_l(x)$ and $g_l(x)$, and to narrow the search of a (t,s)-center z for $p(x)$ to the set of the zeros of the remaining factor of $p^{(l)}(x)$. By continuing recursively, we compute either an (a,ψ)- or an (a,B,ψ)-splitting disc for $p(x)$ or a (t,s)-center for $p(x)$, where $t \geq n/2$. The latter search for a (t,s)-center is actually implicit; it ends with outputing an(a,ψ)-splitting disc for $p(x)$ in $O(\log(n-l))$ recursive steps. To the advantage of this approach, instead of all zeros of $p^{(l)}(x)$ it requires approximation of only one factor of $p^{(l)}(x)$ and the root radii of $p^{(l)}(x+X)$ for some shift value X. This enables a low cost reduction of the original problem of the complete factorization of a polynomial $p(x)$ to two similar problems for its two factors, $F(x)$ and $G(x)$, satisfying the balancing assumption (1.1). By applying this approach, together with the recursive splitting algorithms of Part I, we arrive at Theorem 1.1.

We next specify the resulting factorization algorithm first describing a black box subroutine involved.

Subroutine Split$(v(x), B_v, A)$.

INPUT: a polynomial $v(x)$ of degree n_v, real B_v, and an annulus $A = A(X, r_-, r_+) = \{x : r_- \le \|x - X\| \le r_+\}$ on the complex plane, for positive r_- and r_+ and a complex X.

OUTPUT: Two polynomials, $f^*(x)$ (monic, with all its zeros lying in the disc $D(X, r_-^*)$) and $g^*(x)$ (with all its zeros lying outside the disc $D(X, r_+^*)$), for $r_-^* = qr_-$, $r_+^* = r_+/q$, $r_+/r_- = q^4$, satisfying

$$\|f^*(x)g^*(x) - v(x)\| \le 2^{-B_v}\|v(x)\|. \tag{6.1}$$

ALGORITHM 6.1. *DISC*$(p(x), a, B, c)$.

INPUT: *Polynomial* $p(x) = \sum_{i=0}^{n} p_i x^i$ *of (1.1),* $p_n \ne 0$, *real* a, B, c, ψ, ψ^*, *and* s *(provided that* a *satisfies (1.3),* c, ψ, *and* ψ^* *satisfy (3.1) and (3.3),*

$$B > Cn \log n \tag{6.2}$$

for a sufficiently large constant C, *and* s *satisfies the assumption of Theorem 4.1), and a black-box subroutine Split*$(v(x), B_v, A)$ *specified above.*

OUTPUT: *a) Either an* (a, ψ)*-splitting disc for* $p(x)$ *or*

b) an (a, B, ψ)*-splitting disc for* $p(x)$.

COMPUTATIONS:

Stage 0 (initialization). Write $v(x) = p^{(l-1)}(x)$ *for* $l = \lfloor (3a-2)n \rfloor$, *of (4.1).*

Stage 1. Substitute $n_v = \deg v(x)$ *for* n, ψ_v *for* ψ, *and* ψ_v^* *for* ψ^* *in (3.1) and (3.3) to define* ψ_v *and* ψ_v^*, *apply the subroutine DISC*$(v(x), a, 2B \log s, c)$ *for* c *of (3.1) and (3.3), which outputs an* (a, ψ_v)*- or an* $(a, 2B \log s, \psi_v)$*-splitting disc for* $v(x)$*; denote this disc by* $D = D(C_v, R_v)$. *Shift the origin into its center* C_v *and go to Stage 2.*

Stage 2. Apply Algorithm 3.2 to the input polynomial $p(x)$. *Stop if it outputs an* (a, B, ψ)*- or an* (a, ψ)*-splitting disc for* $p(x)$. *Otherwise stop in* i *recursive steps for the minimal* i *such that the algorithm produces a covering disc* $D(X_i, r_i)$ *with radius* r_i *less than* $(\psi_v - 1)R_v/s$, *where* $(\psi_v - 1)R_v$ *is the width of the annulus produced at Stage 1; in this case invoke the Subroutine Split*$(v(x), B_v, A(C_v, R_v, \psi R_v))$ *with*

$$B_v = C^* B \log s, \tag{6.3}$$

for a sufficiently large constant C^*, *and go to Stage 3.*

Stage 3. Suppose that at Stage 2 Algorithm 3.2 outputs a covering disc $D(X_i, r_i)$. *Write either* $v(x) = f^*(x)$, *if the dilation* $D(X_i, sr_i)$ *intersects the disc* D, *or* $v(x) = g^*(x)$, *otherwise. Then go to Stage 1.*

To see the **correctness of Algorithm 6.1**, observe that according to our policy, at Stage 3, we discard the "wrong" factor of $v(x)$ and stay with the "right" one – we keep a (t, s)-center for $p(x)$ among its zeros. (If we compute an $(a, 2B \log s, \psi_\nu)$-splitting disc for $v(x)$, then a (t, s)-center must lie in this disc, and we compute the desired splitting disc for $p(x)$ based on Proposition 5.2.) The degree of each computed factor of $v(x)$ is bounded from above by a fixed fraction of $\deg v(x)$. Therefore, Algorithm 6.1 must terminate in $O(\log(n - l))$ passes through Stage 3. At termination, it must output an (a, B, ψ)- or an (a, ψ)-splitting disc for $p(x)$. By estimate (19.3) in [39], our bounds (6.1)–(6.3) ensure the relative bounds of order $B \log s$ on the error norms of the computed approximations $f^*(x)$ and $g^*(x)$ to the factors of $v(x)$. Together with the known perturbation theorems (cf., e.g., [40, Theorem 2.7]), this implies that the (t, s)-center for $p(x)$ is closely approximated by the zeros of the selected factors. Furthermore, by Corollary 4.5 and Proposition 5.2, the center C of an $(a, 2B \log s, \psi)$-splitting disc for $v(x)$ computed by Algorithm 6.1 closely approximates a (t, s)-center for $p(x)$, so C itself is a (t, s^*)-center for $p(x)$ where $s^* = s + 1$, say.

To estimate the **cost of the computation** by the algorithm, observe that the entire computation is reduced to the application of Subroutine Split to the auxiliary polynomials $v(x)$ of rapidly decreasing degree, the shifts of the variable x, and the approximation within relative error bound $O(1/n)$ of all root radii of the polynomials $p(x)$ and $v(x)$. The shifts and the root-radii approximation require only $O(n \log^2 n)$ ops performed with $O(n \log n)$-bit precision per recursive step, that is, $O(n \log^3 n)$ ops with $O(n \log n)$-bit precision at all $O(\log n_v)$ recursive steps. This is dominated by the computational cost of all applications of Subroutine Split. By Theorem 2.4 and Remark 2.5, each application involves $O((n_\nu \log n_\nu)(\log^2 n_\nu + \log B_\nu))$ ops performed with $O(B_\nu)$-bit precision.

Now we are ready to prove Theorem 1.1.

PROOF OF THEOREM 1.1. By the above argument, the cost of the computation of an (a, ψ)- or (a, B, ψ)-splitting disc for the polynomial $p(x)$ is dominated by the cost of the subsequent balanced splitting of this polynomial, that is, $O((n \log n)(\log^2 n + \log b))$ ops performed with $O(b)$-bit precision where we choose $b = B \ge n \log n$. Recursive extension of the balanced splitting has depth $O(\log n)$ due to its balancing. The above cost bound applies at every level, and we arrive at Theorem 1.1. □

REMARK 6.2. *As we mentioned, application of recursive Algorithm 3.2 as a block of Algorithm 6.1 can be replaced by a single step of root radii approximation but with* $\psi = 1 + c/n^d$ *and* $\psi^* = 1 + c/n^{d+1}$ *for* $p(x)$ *and with* $\psi_v = 1 + c/n_v^{d_v}$ *and* $\psi_v^* = 1 + c/n_v^{d_v+1}$ *for* $v(x)$, *for a positive* c *and larger* $d > 1$ *and* $d_v > 1$. *In this case, however, we must have* $n^d \ge sn_v^{d_v}$ *to ensure the bounds of (5.1)–(5.3). For* $n_v = l$ *of (4.1), this means* $n^d \ge \lceil (3a-2)n \rceil^{d_v} s > \Theta n^{d_v+1/3}$ *for a fixed positive* Θ *and for* $s = O(n^{1/3})$ *(cf. Remark 4.2). To compute a splitting disc for the polynomial* $p^{(l)}(x)$, *we have to apply Algorithm 6.1 to the polynomials* $p^{(l_i)}(x)$ *where* $l_0 = l$, $l_{i+1} = \lceil (3a-2)l_i \rceil$, $i = 0, 1, \ldots$ *(cf. (4.1)). This would involve* ψ_i*-isolated discs for* $p^{(l_i)}(x)$ *with* $\psi_i - 1$ *of the order of* $1/n^{d-i/3}$. *We must generally deal with the number of recursive steps* i *of the order of* $\log n$, *which means that the exponent* d *must also be of this order. That is, the considered modification of Algorithms 3.2 and 6.1 required splitting* $p(x)$ *over an annulus with a relative width of the order of* $1/n^{\log n}$. *To yield this splitting, we must extend Theorem 2.4 and the lifting/descending construction in [33]. Then the resulting estimated arithmetic time-cost of factorization and rootfinding would have increased a little (at least by the factor of* $\log n$*), but what is much worse, the computation of the required splitting would have involved unreasonably long bit-precision, of the order of* $n^{\log n}$, *and, therefore, dramaticly blowing up of the Boolean (bit-operation) cost.*

7. REFERENCES

[1] L. Ahlfors, *Complex analysis*, McGraw-Hill, New York, 1979.

[2] L. Blum, F. Cucker, M. Shub, S. Smale, *Complexity and Real Computations*, Springer, New York, 1997.

[3] R. P. Brent, F. G. Gustavson, D. Y. Y. Yun, Fast solution of Toeplitz systems of equations and computation of Padé approximations, *J. Algorithms*, **1**, 259-295, 1980.

[4] D. Bini and V. Y. Pan, *Polynomial and Matrix Computations, v. 1: Fundamental Algorithms*, Birkhäuser, Boston, 1994.

[5] D. Bini and V. Y. Pan, *Polynomial and Matrix Computations, v. 2: Fundamental and Practical Algorithms*, Birkhäuser, Boston, to appear.

[6] J. R. Bunch, Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM J. Sci. Stat. Comput.*, **6, 2**, 349-364, 1985.

[7] D. Coppersmith and C.A. Neff, Roots of a Polynomial and Its Derivatives, *Proc. 5-th Ann. ACM-SIAM Symp. on Discrete Algorithms*, 271–279, ACM Press, New York, and SIAM Publications, Philadelpha, 1994.

[8] L. M. Delves, J. N. Lyness, A numerical method for locating zeros of an analytic functions, *Math. Comp.*, **21**, 543-560, 1967.

[9] I. Z. Emiris, A. Galligo, H. Lombardi, Numerical Univariate Polynomial GCD, *Proc. of AMS-SIAM Summer Seminar: Mathematics of Numerical Analysis: Real Number Algorithms, (Park City, Utah, 1995), Lectures in Applied Math.*, **32**, 323-343, Amer. Math. Society, Providence, Rhode Island, 1996.

[10] I. Z. Emiris, A. Galligo, H. Lombardi, Certified approximate polynomial gcds, *J. Pure and Applied Algebra*, **117/118**, 229-251, 1997.

[11] A. Gel'fond, *Differenzenrechnung, Deutsher Verlag Der Wissenschaften*, Berlin, 1958 (Russian edition: Moscow, 1952).

[12] A. A. Grau, The simultaneous improvement of a complete set of approximate factors of a polynomial, *SIAM J. of Numer. Analysis*, **8**, 425-438, 1971.

[13] W. B. Gragg, The Padé table and its relation to certain algorithms of numerical analysis, *SIAM Review*, **14, 1**, 1-62, 1972.

[14] P. Kirrinnis, Fast computation of numerical partial fraction decompositions and contour integrals of rational functions, *Proc. Inter. Symp. on Symb. and Algebraic Comput. (ISSAC 92)*, (Paul S. Wang editor), 16–26, ACM Press, New York, 1992.

[15] P. Kirrinnis, Polynomial factorization and partial fraction decomposition by simultaneous Newton's iteration, *J. of Complexity*, 14, 3, pp. 378–444, 1998.

[16] D. Kapur and Y. N. Lakshman, Elimination methods: An introduction, in *Symbolic and Numerical Computation for Artificial Intelligence* (B. Donald, D. Kapur, and J. Mundy, editors), 45–89. Academic Press, New York, 1992.

[17] J. M. McNamee, Bibliography on roots of polynomials, *J. Comp. Appl. Math.*, **47**, 391-394, 1993.

[18] J. M. McNamee, A supplementary bibliography on roots of polynomials, *J. Computational Applied Mathematics*, **78, 1**, 1997, also http://www.elsevier.nl/homepage/sac/cam/mcnamee/index.html.

[19] M. Mignotte, An inequality about factors of polynomials, *Math. Comp.*, **28**, 1153-1157, 1974.

[20] B. Mourrain and V. Y. Pan, Asymptotic acceleration of solving polynomial systems, *Proc. 27th Ann. ACM Symp. on Theory of Computing*, 488–496, ACM Press, New York, May 1998.

[21] B. Mourrain and V. Y. Pan, Multivariate polynomials, duality and structured matrices, *J. of Complexity*, **16, 1**, 110–180, 2000.

[22] C. A. Neff, Specified precision polynomial root isolation is NC, *J. of Computer and System sciences*, **48**, 429–463, 1994.

[23] C. A. Neff and J. H. Reif, An $O(n^{l+\epsilon})$ algorithm for the complex root problem, in *Proc. 35th Ann. IEEE Symp. on Foundations of Computer Science*, 540–547, IEEE Computer Society Press, Los Alamitos, California, 1994.

[24] C. A. Neff and J. H. Reif, An efficient algorithm for the complex roots problem, *J. of Complexity*, **12**, 81–115, 1996.

[25] V. Y. Pan, Sequential and parallel complexity of approximate evaluation of polynomial zeros, *Computers & Math. (with Applications)*, **14, 8**, 591–622, 1987.

[26] V. Y. Pan, *Optimal (up to polylog factors) sequential and parallel algorithms for approximating complex polynomial zeros*, Proc. 27th Ann. ACM Symp. on Theory of Computing, pp. 741–750, ACM Press, New York, May, 1995.

[27] V. Y. Pan, Deterministic improvement of complex polynomial factorization based on the properties of the associated resultant, *Computers & Math. (with Applications)*, **30, 2**, 71-94, 1995.

[28] V. Y. Pan, *Optimal and nearly optimal algorithms for approximating polynomial zeros*, Computers & Math. (with Applications), 31, 12, pp. 97–138, 1996.

[29] V. Y. Pan, *Solving a polynomial equation: Some history and recent progress*, SIAM Review, 39, 2, pp. 187–220, 1997.

[30] V. Y. Pan, *Approximate polynomial gcds, Padé approximation, polynomial zeros, and bipartite graphs*, Proc. 9th Ann. ACM-SIAM Symp. on Discrete Algorithms, pp. 68–77, ACM Press, New York, and SIAM Publications, Philadelphia, 1998.

[31] V. Y. Pan, *Approximating complex polynomial zeros: modified quadtree (Weyl's) construction and improved Newton's iteration*, J. of Complexity, 16, 1, pp. 213–264, 2000.

[32] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Birkhäuser/Springer, 2001, to appear.

[33] V. Y. Pan, Perturbation of Padé approximation, Preprint, 2000.

[34] V. Y. Pan, Computation of Approximate Polynomial GCD's and an Extension, *Information and Computation*, to appear.

[35] V. Y. Pan and Z. Q. Chen, *The Complexity of the Matrix Eigenproblem*, Proc. 31st Annual ACM Symposium on Theory of Computing, pp. 507–516, ACM Press, New York, 1999.

[36] G. Pólya, G. Szegö, *Aufgaben und Lehrsätze aus der Analysis*, Verlag Von Julius Springer, Berlin, 1925.

[37] J. Renegar, On the worst-case arithmetic complexity of approximating zeros of polynomials, *J. Complexity*, **3, 2**, 90–113, 1987.

[38] A. Schönhage, *Asymptotically fast algorithms for the numerical multiplication and division of polynomials with complex coefficients*, Proc. EUROCAM, Marseille, Lecture Notes in Computer Science, 144, pp. 3–15, Springer, Berlin, 1982.

[39] A. Schönhage, *The fundamental theorem of algebra in terms of computational complexity*, Math. Dept., University of Tübingen, Tübingen, Germany, 1982.

[40] A. Schönhage, Quasi-gcd Computations, *J. of Complexity*, **1**, 118–137, 1985.

[41] S. Smale, The fundamental theorem of algebra and complexity theory, *Bull. Amer. Math. Soc.*, **4, 1**, 1-36, 1981.

[42] S. Smale, On the efficiency of algorithms of analysis, *Bulletin of the American Mathematical Society*, **13, 2**, 87-121, 1985.

APPENDIX

A. EXTENSION TO SPLITTING OVER ANY CIRCLE

By the initial scaling of the variable, we may move the zeros of a given polynomial into the unit disc $D(0,1)$. Therefore, it is sufficient to consider splitting of a polynomial p of (1.1) (within a fixed error tolerance ϵ) over any disc $D(X,r)$, with X and r satisfying the bounds $r > 0$ and

$$r + |X| \leq 1. \tag{A.1}$$

To extend the splitting respectively, we shift and scale the variable x and estimate the new relative error norm bound $\tilde{\epsilon}$ as a function in ϵ, X and r. The following result relates ϵ and $\tilde{\epsilon}$.

Proposition A.1. *Let relations (1.11) and (A.1) hold. Write*

$$y = rx + X, \tag{A.2}$$

$$\begin{aligned} \tilde{p}(y) &= \sum_{i=0}^{n} \tilde{p}_i y^i = \tilde{p}(rx+X) = q(x), \\ p(x) &= q(x)/\|q(x)\|, \end{aligned} \tag{A.3}$$

$$\begin{aligned} \widetilde{F}^*(y) &= \widetilde{F}^*(rx+X) = F^*(x)r^k, \\ \widetilde{G}^*(y) &= \widetilde{G}^*(rx+X) = G^*(x)/(\|q(x)\|r^k), \\ \Delta(x) &= p(x) - F^*(x)G^*(x), \\ \widetilde{\Delta}(y) &= \tilde{p}(y) - \widetilde{F}^*(y)\widetilde{G}^*(y). \end{aligned}$$

Then (A.2) maps the disc $D(0,1) = \{x : |x| \leq 1\}$ onto the disc $D(X,r) = \{y : |y - X| \leq r\}$; moreover,

$$\begin{aligned} \|\widetilde{\Delta}(y)\| &\leq \|\Delta(x)\| \cdot ((1+|X|)/r)^n \cdot \|\tilde{p}(y)\| \\ &\leq \|\Delta(x)\| \cdot ((2-r)/r)^n \cdot \|\tilde{p}(y)\|. \end{aligned} \tag{A.4}$$

Proof. Clearly, (A.2) maps the disc $D(X,r)$ as stated. To prove (A.4), first note that $\Delta(x) = \Delta(\frac{y-X}{r}) = \widetilde{\Delta}(y)/\|q(x)\|$. Therefore,

$$\|\widetilde{\Delta}(y)\| = \|\Delta(\tfrac{y-X}{r})\| \cdot \|q(x)\|. \tag{A.5}$$

Combine the relations $1 \leq \|(y-X)^i/r^i\| = (1+|X|)^i/r^i$, for $i = 0, 1, \ldots$, with (A.1) and deduce that

$$\begin{aligned} \|\Delta\left(\tfrac{y-X}{r}\right)\| &\leq \|\Delta(x)\| \cdot \max_i \left(\tfrac{\|(y-X)^i\|}{r^i}\right) \\ &= \|\Delta(x)\| \left(\tfrac{1+|X|}{r}\right)^n \\ &\leq \|\Delta(x)\| \left(\tfrac{2-r}{r}\right)^n. \end{aligned} \tag{A.6}$$

On the other hand, having $q(x) = \tilde{p}(rx+X)$ and $\|(rx+X)^i\| = (r+|X|)^i$ for $i = 0, 1, \ldots$, we deduce that

$$\|q(x)\| = \|\tilde{p}(rx+X)\| = \left\|\sum_{i=0}^{n} \tilde{p}_i(rx+X)^i\right\| \leq \sum_{i=0}^{n} |\tilde{p}_i|(r+|X|)^i.$$

Due to (A.1), it follows that

$$\|q(x)\| \leq \sum_{i=0}^{n} |\tilde{p}_i| = \|\tilde{p}(y)\|.$$

Combine the latter bound with (A.5) and (A.6) to obtain (A.4). □

B. ERROR ESTIMATES FOR RECURSIVE SPLITTING

Suppose that we recursively split each approximate factor of p over the boundary circle of some well-isolated disc until we arrive at the factors of the form $(ux+v)^d$. This gives us an approximate factorization

$$p^* = p^*(x) = \prod_{j=1}^{n} (u_j x + v_j). \tag{B.1}$$

Let us estimate the norm of the residual polynomial $\Delta^* = p^* - p$. We start with an auxiliary result.

Proposition B.1. *[39, §5]. Let*

$$\Delta_k = |p - f_1 \ldots f_k| \leq k\epsilon|p|/n, \tag{B.2}$$

$$\Delta = |f_1 - fg| \leq \epsilon_k |f_1|, \tag{B.3}$$

for some nonconstant polynomials $f_1, \ldots, f_k, f$ and g and for

$$\epsilon_k \leq \epsilon|p|/(n \prod_{i=1}^{k} |f_i|). \tag{B.4}$$

Then

$$|\Delta_{k+1}| = |p - fgf_2 \ldots f_k| \leq (k+1)\epsilon|p|/n. \tag{B.5}$$

PROOF. $\Delta_{k+1} = |p - f_1 \dots f_k + (f_1 - fg)f_2 \dots f_k| \le \Delta_k + \Delta|f_2 \dots f_k|$. Substitute (B.2)–(B.4) and deduce (B.5). □

Write $f_1 = f$, $f_{k+1} = g$. Then (B.5) turns into (B.2) for k replaced by $k+1$. Now split one of the factors f_i as in (B.3), apply Proposition B.1, and recursively split p into factors of smaller degrees until we arrive at (B.1), where

$$|\Delta^*| = |p^* - p| \le \epsilon|p|. \tag{B.6}$$

Let us call this computation **Recursive Splitting Process** provided that it starts with $k = 1$ and $f_1 = p$ and ends with $k = n$.

PROPOSITION B.2. *[39]. Performing Recursive Splitting Process for a positive $\epsilon \le 1$, it is sufficient to choose ϵ_k in (B.3) satisfying*

$$\epsilon_k \le \epsilon/(n2^{n+1}) \tag{B.7}$$

for all k to support (B.2) for all $k = 1, 2, \dots, n$.

PROOF. We prove (B.2) for all k by induction on k. Clearly, the bound holds for $k = 1$. Therefore, it remains to deduce (B.5) from (B.2) and (B.7) for any k. By first applying Proposition 4.1 and then (B.2), we obtain that

$$\prod_{i=1}^{k} |f_i| \le 2^n \left| \prod_{i=1}^{k} f_i \right| \le 2^n (1 + k\epsilon/n)|p| \le 2^{n+1}|p|$$

for $k \le n, \epsilon \le 1$. Consequently, (B.7) ensures (B.4), and then (B.5) follows by Proposition B.1. □

C. MODIFICATIONS OF THE DESCENDING PROCESS

Consider modifications of the descending stage of Algorithm 2.1 of Part I based on either or both of the two following equations applied for all j:

$$F_{u-j}(x) = \gcd(q_{u-j}(x), F_{u-j+1}(x^2)),$$
$$G_{u-j}(x) = \gcd(q_{u-j}(x), G_{u-j+1}(x^2)), \quad j = 1, \dots, u.$$

Here and hereafter, $\gcd(u(x), v(x))$ denotes the monic **g**reatest **c**ommon **d**ivisor (gcd) of the two polynomials $u(x)$ and $v(x)$.

In this modification of Algorithm 2.1, Padé computation is replaced by the polynomial gcd computation. This produces the same output as in Algorithm 2.1 if we assume infinite precision of computing. The approach was originally introduced in the proceedings paper [26] but in its journal version [28] was replaced by the one based on Padé computation. The replacement enabled more direct control over the propagation of the perturbation errors (cf. Theorem 3.4 in Part I), although both approaches can be made computationally equivalent because both Padé and gcd computations can be reduced to the same Toeplitz linear system of equations (cf. [3], [4], [32]).

The gcd approach, however, may lead into a trap if one tries to solve the gcd problems based on the fast Euclidean algorithm (cf. Algorithm 5.1a in [4, chapter 1] or Algorithm 2 in [32]). In this case, each descending step (2.4) in Part I is replaced by a recursive Euclidean process, prone to the severe problems of numerical stability (cf. [9], [10], and [40]) and to blowing up the precision of the computations and the Boolean cost. In particular, the paper [24] has fallen into this trap. The authors apply the fast Euclidean algorithm in the gcd version of the descending process, reproduced from [26], but unfortunately, the analysis hinges on the invalid assumption that the value $\delta = \psi - 1$ exceeds a fixed positive constant (ψ^2 being the relative width of the basic annulus for splitting a polynomial q_{u-j}). This assumption is satisfied only for the polynomials q_{u-j} computed at a few last lifting steps, that is, for $j = u - O(1)$ but not for $j = 0, 1, \dots, u/2$ (say). Thus, the analysis presented in [24] applies to only few first descending steps, and the Boolean cost of performing all other steps remains unbounded. Furthermore, this flaw is not easy to fix; clearly it cannot be fixed based on the techniques in [24].

D. SOME RELATED WORKS

The study of polynomial rootfinding is related to various areas of pure and applied mathematics as well as the theory and practice of computing and has huge bibliography [5], [17], [18], [29]. We focus on one important aspect of this study, that is, the computational complexity of the solution under the arithmetic and Boolean (bit-operation) models. The modern interest to this aspect of the study is due to [39], [41], and [42], and major progress was obtained quite recently. Nearly optimal solution algorithms appeared in [26], [28]. They rely on the recursive balanced splitting of an input polynomial $p = p(x)$ into the product of two factors of balanced degrees (that is, neither the ratio of the degrees nor its reciprocal can exceed a fixed constant).

[17], [18] is a good source for the bibliography on the preceding works; the unpublished manuscript [39] is an important landmark work but is sparse in citations.

In [15] and [39] algorithms for splitting were studied extensively, assuming a sufficiently high relative width of the basic zero-free annulus, that is, assuming higher isolation of the zeros of the factors from each other. No balancing of the degrees of the factors was achieved. Further improvements by the order of magnitude were due to relaxing the assumption of the isolation (by reversing Graeffe's lifting process with using Padé approximation) [26], [28], and to achieving balanced splitting [7], [11], [23], [26]. The techniques of the papers [15], [23] and [39] are more important and lasting longer than the computational complexity estimates. [15] reached the same bound on the Boolean (bit-operation) cost as [26], [28] but only under the weird requirement of blowing up the precision of computing to the order of $(1 + \frac{1}{r \log n})n^2$ bits, r being the minimum distance between the distinct roots. Otherwise the algorithms in [15] and [39] support the arithmetic and Boolean cost bounds that exceed the bounds in [26], [28] by roughly the factors of n^2 and n, respectively.

Theorems in [7], [11], and [23] contributed highly important advanced technique for balancing the degrees of the factors produced by splittings, but contrary to the claim in [23], defined no construction that would have supported nearly optimal complexity estimates. Theorems in [7] and [11] are on the complexification of Rolle's theorem, not on rootfinding, whereas the construction in [23] is quite preliminary. [23] relies on a more straightforward recursive splitting of higher order derivatives of $p(x)$, avoided in [26], [28], and this already means a waste of the factor of n^δ for a positive δ in the parallel and sequential time bounds. Furthermore, the construction in [23] does not include the recursive process of Algorithm 3.2 and relies on the approximation of the root radii with a higher precision. In our Remark 6.2 of Part II we note the resulting dramatic increase of the precision and the Boolean computational cost. The algorithm in [23] was also “simplified” by ignoring the problem of massive clusters, in particular neither (a, B, ψ)-splitting discs nor any alternative for them were introduced.

The algorithms in [26] and [28] rely on the constructions similar to those presented here and in particular involve the computation of a single factor of a higher order derivative $p^{(l)}(x)$ in each recursive step. The algorithms, however, assume a higher precision of computation, to ensure that all the zeros of $p^{(l)}$ are perturbed very little. Now we relax this excessively strong requirement. *We only bound the errors of the recursive factorization*, which is a much weaker requirement and which we found sufficient for our goal of computing an (a, ψ)- or an (a, B, ψ)-splitting disc for $p(x)$. The construction in [26], [28] is also more complicated apart from its reliance on a less effective splitting algorithm. All this resulted in the requirement of a higher computational precision and in the estimates of the order of $(b + n)n^2$ for the Boolean cost of the complete factorization versus the order of $(b + n)n$ in Theorem 1.1 (in both cases up to polylog factors).

The Boolean cost bound in [26], [28] is still optimal for the rootfinding for the worst case input polynomial because of the precision growth required for the worst case rootfinding versus factorization, and our present algorithm also yields this bound. The bound, however, is by the factor n inferior to ours in the case of polynomials with well-conditioned zeros and also for the auxiliary stage of computing numerical factorization into linear factors, which is of independent interest.

The paper [24] very closely follows the earlier work [26] (which Neff refereed for the ACM STOC in December 1994) but complements it with the Boolean complexity analysis of the descending process. The analysis, however, falls apart because of an irreparable flaw (see Appendix C).