Theoretical Computer Science 409 (2008) 292–310

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Complexity of real root isolation using continued fractions

Vikram Sharma

Max-Planck Institut für Informatik, Saarbrücken, 66123, Germany

ARTICLE INFO

Keywords:
Polynomial real root isolation
Continued fractions
The Descartes' rule of signs
Davenport–Mahler bound

ABSTRACT

In this paper, we provide polynomial bounds on the worst case bit-complexity of two formulations of the continued fraction algorithm. In particular, for a square-free integer polynomial of degree n with coefficients of bit-length L, we show that the bit-complexity of Akritas' formulation is $\widetilde{O}(n^8L^3)$, and the bit-complexity of a formulation by Akritas and Strzeboński is $\widetilde{O}(n^7L^2)$; here $\widetilde{O}$ indicates that we are omitting logarithmic factors. The analyses use a bound by Hong to compute the floor of the smallest positive root of a polynomial, which is a crucial step in the continued fraction algorithm. We also propose a modification of the latter formulation that achieves a bit-complexity of $\widetilde{O}(n^5L^2)$.

1. Introduction

A fundamental task in computer algebra is **real root isolation**, i.e., given a polynomial $A(X)$ with real numbers as coefficients, compute disjoint intervals with rational endpoints such that each contains exactly one real root of $A(X)$, and together they contain all the real roots of $A(X)$. In this paper, we assume $A(X)$ is square-free and has degree n. [1]

The **continued fraction algorithm**[2] for real root isolation, i.e., an algorithm which utilizes the continued fraction expansion of the real roots of a polynomial for isolating them, was first proposed by Vincent [34]; see [8] for detailed historic information of the algorithm. Vincent observed that if the polynomial $A(X)$ is recursively transformed as $A_{i+1}(X) := X^n A_i(a_i + 1/X)$, where $A_0(X) := A(X)$, $a_0 \in \mathbb{N}_{\geq 0}$, and $a_i \in \mathbb{N}_{>0}$, then eventually the polynomial has at most one sign variation in its coefficients. Along with the Descartes' rule of signs (see Proposition 2.1), we get an algorithm for real root isolation; see Section 2 for the details. The quantity a_i is the floor of some positive real root of the polynomial $A_i(X)$. Different ways of computing a_i yield us different formulations of the continued fraction algorithm. Vincent computed a_i by performing Taylor shifts by one on $A_i(X)$, but this leads to an exponential worst case running time. To overcome this drawback, Akritas [4] suggested that to compute a_i we should compute a lower bound b on the smallest positive root of $A_i(X)$; if $b \geq 1$ then perform a Taylor shift on $A_i(X)$ by b, instead of by one as done by Vincent, and repeat this until the root that has a_i as its floor is in the unit interval; for computing the lower bound, Akritas used Cauchy's bound [5, p. 350]. The algorithm by Collins and Akritas [11], though not a continued fraction algorithm, was also proposed to overcome the exponential drawback of Vincent's algorithm.

In practice, Akritas' formulation of the continued fraction algorithm almost always outperforms the algorithm by Collins and Akritas [31]. But what is interesting about the formulation is that, unlike the latter algorithm, it utilizes the distribution of the real roots of the polynomial to isolate them; the advantage of this approach is evident (see [31, Tab. 1]) when we isolate the real roots of Mignotte's polynomials [22], where it is known that the approach of Collins and Akritas is not

E-mail address: vsharma@mpi-inf.mpg.de.
URL: http://www.mpi-inf.mpg.de/~vsharma.

[1] The results presented here are based upon [28,29].

[2] Recently [2] it has been called the Vincent–Akritas–Strzeboński method. However, for the purposes of complexity analysis we need to consider each formulation of this algorithm by the three authors separately, and hence we do not use this name.

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.09.017

efficient [15, Thm. 3.6]. Another advantage, shared by all formulations of the continued fraction algorithm, is that the rational approximations computed to the real roots are their continued fraction expansion, which is the best approximation one can expect for a given magnitude of the denominator of the fraction (see [18, Sec. 6]).

Despite these advantages, the known polynomial bounds on the worst case complexity of Akritas' formulation of the continued fraction algorithm have gaps in their proofs. Akritas has claimed an $\widetilde{O}(n^5L^3)$ bound on the worst case bit-complexity of his formulation, using classical arithmetic for implementing Taylor shifts [20]; if we use asymptotically fast Taylor shift [35, Thm. 2.4] then his bound can be improved to $\widetilde{O}(n^4L^2)$. His analysis, however, has two drawbacks: first, he assumes the **ideal Positive Lower Bound (**PLB**) function**, i.e., a function that can determine whether a polynomial has positive real roots, and if there are such roots then returns the floor of the smallest positive root of the polynomial; and second, his analysis does not account for the increased coefficient size of $A_{i+1}(X)$ after performing Taylor shift by b on $A_i(X)$. In practice, we never use the ideal PLB function because of its prohibitive cost (intuitively it is almost equivalent to doing real root isolation). Instead we use functions that are based upon efficient upper bounds on positive roots of a polynomial (e.g. [7,17,19,30]); to compute a lower bound on the smallest positive root of $A_i(X)$, we compute an upper bound on the positive roots of the polynomial $X^nA_i(1/X)$ and take its inverse. Based upon such functions, a polynomial bound on the worst case complexity of Akritas' formulation of the continued fraction algorithm appears in [28, Chap. 3]; instead of the ideal PLB function the analysis in [28, Chap. 3] assumes a function that uses Zassenhaus' bound [36, Lem. 6.5, p. 147] for computing upper bounds on positive roots of a polynomial. Further improvement was obtained in [29] using a bound by Hong [17] instead of Zassenhaus' bound. Both the results, however, have a gap in their proofs. To understand the gap, suppose that $a_1, \ldots, a_\ell$ are in $\mathbb{N}_{\geq 1}$ and we recursively define polynomials $B_i(X) := B_{i-1}(X + a_i)$, where $B_0(X) := A(X)$. Then the analyses [28, Chap. 3] and [29] assume that the complexity of computing $B_{\ell+1}(X)$ is bounded by the complexity of computing $A(X + \sum_{i=1}^{\ell} a_i)$; this is not true, because the bit-length of the former is bounded by $O(\log \|A\|_\infty + n \sum_{i=1}^{\ell} \log a_i)$, whereas that of the latter is bounded by $O(\log \|A\|_\infty + n \log \sum_{i=1}^{\ell} a_i)$.

In this paper, we remedy this error, and also improve by a factor of n upon the worst case tree size derived in [29]. More precisely, for a square-free integer polynomial of degree n with L-bit coefficients, we derive in Section 3 a worst case bound of $\widetilde{O}(n^2L)$ on the tree size of Akritas' formulation of the continued fraction algorithm using the bound by Hong [17] for computing a lower bound on the smallest positive root. The worst case complexity of the formulation after compensating for the above mentioned gap, however, is $\widetilde{O}(n^8L^3)$; this is derived in Section 4. The crucial component for bounding the size of the recursion tree of the formulation, without assuming the ideal PLB function, is the tightness of the lower bounds on the positive real roots of the polynomial; this is the subject that we treat in Section 2, where we also give the details of Akritas' formulation of the continued fraction algorithm.

A recent formulation of the continued fraction algorithm by Akritas and Strzeboński [3] outperforms the earlier formulation by Akritas. The former algorithm proceeds similarly to the latter algorithm, i.e., it computes a lower bound b on the smallest positive root of the polynomial $A_i(X)$, and if $b \geq 1$ it computes $A_i(b(X + 1))$, instead of $A_i(X + b)$. In Section 5, we derive an $\widetilde{O}(n^7L^2)$ bound on the worst case complexity of this recent formulation. We further modify the formulation of Akritas and Strzeboński to obtain a formulation that has a worst case complexity of $\widetilde{O}(n^5L^2)$, and thus matches the known worst case complexity result, assuming classical arithmetic for implementing Taylor shifts [20], for the Collins and Akritas' algorithm (see e.g., [9, Sec. 10.2], [15]). Under the same assumption on Taylor shifts, this result also matches the *expected complexity* of Akritas' formulation. Without the assumption, though, the expected complexity of Akritas' formulation is $\widetilde{O}(n^3L)$ [31,32]; instead of the ideal PLB function, the analyses in both the references assume that the set of real algebraic numbers of degree greater than or equal to three follows Gauss–Kuzmin's distribution and Khinchin's law [18, Chap. 3], i.e., in the continued fraction expansion of such real algebraic numbers the expected bit size of the partial quotients is a constant.

We begin with introducing some notation, borrowed from [36, Ch. 15], on the theory of continued fractions.

1.1. Notation

A continued fraction is a possibly infinite expression of the form

$$q_0 + \cfrac{p_1}{q_1 + \cfrac{p_2}{q_2 + \cfrac{p_3}{q_3 + \cdots}}},$$

where $p_i \in \mathbb{N}$ is the ith partial numerator and $q_i \in \mathbb{N}$ is the ith partial denominator. For the ease of writing, we express it as $[q_0, \frac{p_1}{q_1}, \frac{p_2}{q_2}, \frac{p_3}{q_3} \ldots]$. If all the p_i's are one then the continued fraction is called an **ordinary continued fraction** (also called simple continued fraction, or regular continued fraction); for convenience again, we express it as $[q_0, q_1, q_2, \ldots]$.

If P_i/Q_i denotes the finite continued fraction $[q_0, \frac{p_1}{q_1}, \ldots, \frac{p_i}{q_i}]$ then we have the following recurrence

$$P_i = p_iP_{i-2} + q_iP_{i-1} \text{ and } Q_i = p_iQ_{i-2} + q_iQ_{i-1}, \tag{1}$$

where $P_{-1} := 1$, $P_0 := q_0$ and $Q_{-1} := 0$, $Q_0 := 1$. Moreover, from [36, p. 463] we know that

$$|P_iQ_{i-1} - P_{i-1}Q_i| = p_1p_2 \ldots p_i. \tag{2}$$

Furthermore, with the finite continued fraction $[q_0, \frac{p_1}{q_1}, \ldots, \frac{p_i}{q_i}]$ we can associate the Möbius transformation

$$M(X) := \frac{P_{i-1} + P_i X}{Q_{i-1} + Q_i X};$$

from (2) it is clear that the Möbius transformation associated with an ordinary continued fraction is unimodular. Let I_M be the interval with endpoints $M(\infty) = P_i/Q_i$ and $M(0) = P_{i-1}/Q_{i-1}$.

In the following sections, we use log to represent $\log_2$, and $O(\cdot)$ means that we are omitting logarithmic factors.

2. Continued fraction algorithm

Given a polynomial $A(X) = a_n X^n + a_{n-1}X^{n-1} + \cdots + a_0$, $a_i \in \mathbb{R}$, let $\mathtt{Var}(A)$ represent the number of sign changes (from positive to negative and vice versa) in the sequence $(a_n, a_{n-1}, \ldots, a_0)$, after removing all zeros from it.

The first crucial component of the continued fraction algorithm is the Descartes' rule of signs:

Proposition 2.1. *Let $A(X)$ be a polynomial with real coefficients. Then the number of positive real roots of $A(X)$ counted with multiplicities is smaller than $\mathtt{Var}(A)$ by a non-negative even number.*

See [21] for a proof with careful historic references. Because $\mathtt{Var}(A)$ exceeds the number of positive roots by a non-negative even number, the Descartes' rule of signs yields the exact number of positive roots when $\mathtt{Var}(A)$ is 0 or 1.

The second crucial component is a procedure PLB(A) that takes as input a polynomial $A(X)$ and returns a lower bound on the smallest positive root of $A(X)$, if such a root exists; our implementation of this procedure, however, will only guarantee a weaker inequality, namely a lower bound on the smallest real part amongst all the roots of $A(X)$ in the positive half plane.

Given these two components, the continued fraction algorithm for isolating the real roots of a square-free input polynomial $A_{\text{in}}(X)$ uses a recursive procedure CF(A, M) that takes as input a polynomial $A(X)$ and a Möbius transformation $M(X) = \frac{pX+q}{rX+s}$, where $p, q, r, s \in \mathbb{N}$ and $ps - rq \neq 0$. The interval I_M associated with the transformation $M(X)$ has endpoints p/r and q/s. The relation among $A_{\text{in}}(X)$, $A(X)$ and $M(X)$ is the following:

$$A(X) = (rX + s)^n A_{\text{in}}(M(X)). \tag{3}$$

Given this relation, the procedure CF(A, M) returns a list of isolating intervals for the roots of $A_{\text{in}}(X)$ in I_M. To isolate all the positive roots of $A_{\text{in}}(X)$, initiate CF(A, M) with $A(X) = A_{\text{in}}(X)$ and $M(X) = X$; to isolate the negative roots of $A_{\text{in}}(X)$, initiate CF(A, M) on $A(X) := A_{\text{in}}(-X)$ and $M(X) = X$, and swap the endpoints of the intervals returned while simultaneously changing their sign.

The procedure CF(A, M) is as follows:

Procedure CF(A, M)
Input: A square-free polynomial $A(X) \in \mathbb{R}[X]$ and a Möbius transformation $M(X)$ satisfying (3).
Output: A list of isolating intervals for the roots of $A_{\text{in}}(X)$ in I_M.

1. **If** $A(0) = 0$ **then**
   Output the interval $[M(0), M(0)]$.
   $A(X) := A(X)/X$; **return** CF(A, M).
2. **If** $\mathtt{Var}(A) = 0$ **then return**.
3. **If** $\mathtt{Var}(A) = 1$ **then**
   Output the interval I_M and **return**.
4. $b := \text{PLB}(A)$.
5. **If** $b \geq 1$ **then**
   $A(X) := A(X + b)$, $M(X) := M(X + b)$.
6. $A_R(X) := A(1 + X)$ and $M_R(X) := M(1 + X)$.
7. CF(A_R, M_R).
8. **If** $\mathtt{Var}(A_R) < \mathtt{Var}(A)$ **then**
9. $A_L(X) := (1 + X)^n A\left(\frac{1}{1+X}\right)$,
   $M_L(X) := M\left(\frac{1}{1+X}\right)$.
10. **If** $A_L(0) = 0$ **then** $A_L(X) := A_L(X)/X$.
11. CF(A_L, M_L).

Vincent's formulation is obtained by removing lines 4 and 5 from the procedure CF, and hence it is not immediately apparent if the formulation by Akritas is more efficient than Vincent's.

We now give the details of the positive lower bound function used in the algorithm above.

2.1. Lower bounds on positive roots

Given a polynomial $A(X) = \sum_{i=0}^{n} a_i X^i$, $a_0 \neq 0$, one way to compute PLB(A) is to take the inverse of an upper bound on the largest positive root of the polynomial

$$R(A)(X) := X^n A(1/X).$$

To compute an upper bound we use the following bound by Hong [17]

$$H(A) := 2 \max_{a_i<0} \min_{a_j>0, j>i} \left| \frac{a_i}{a_j} \right|^{1/(j-i)}.$$

To be more precise, $H(A)$ is an upper bound on the absolute positiveness of $A(X)$, i.e., a bound such that the evaluation of the polynomial and *all* its derivatives at any point larger than the bound is strictly positive; for univariate polynomials this means the bound by Hong is an upper bound on P_A, the maximum amongst the positive roots of the polynomial and its derivatives. Moreover, Hong showed that (see [17, Thm. 5.3])

$$P_A < H(A) \leq \frac{2n}{\ln 2} P_A.$$

But from repeated applications of Gauss–Lucas theorem [23, Thm. 2.2.1, p. 93] we know that all the roots of the derivatives of $A(X)$ are contained inside the convex hull of its roots, and hence if R_A denotes the largest real part amongst all the roots of $A(X)$ in $\Re(z) > 0$ then it follows that

$$P_A < H(A) \leq \frac{2n}{\ln 2} R_A. \tag{4}$$

Clearly, $H(A)$ cannot be always computed exactly using rational arithmetic. Instead, we use a procedure $U(A)$, similar to that suggested by Akritas [5, p. 350], which computes an upper bound on the positive roots of $A(X) \in \mathbb{R}[X]$.

PROCEDURE U(A)
INPUT: An integer polynomial $A(X) = \sum_{i=0}^{n} a_i x^i$, $a_i \in \mathbb{R}$, and $a_n > 0$.
OUTPUT: An upper bound on the positive real roots of $A(X)$.
1. $q' := -\infty$.
2. **For** $a_i < 0, 0 \leq i \leq n-1$, do:
   $q'' := \infty$.
   For $a_j > 0, i < j \leq n$, do:
   $p := \lfloor \log |a_i| \rfloor - \lfloor \log |a_j| \rfloor - 1$.
   $q'' := \min(q'', \lfloor p/(j-i) \rfloor + 2)$.
   $q' := \max(q', q'')$.
3. **Return** $2^{q'+1}$.

Remark 2.2. If $A(X)$ is an integer polynomial with coefficients of bit-length L_A then the cost of computing $U(A)$ is $\widetilde{O}(n^2 L_A)$, because the most expensive operation in the loop is computing the floor of the coefficients, which can be done in $O(L_A)$ time, and the loop iterates n^2 times.

We have the following relation between $U(A)$ and $H(A)$:

Lemma 2.3.

$$\frac{U(A)}{4} < H(A) < U(A).$$

Suppose $H(A) = 2|a_i/a_j|^{1/(j-i)}$. Let $p := \lfloor \log |a_i| \rfloor - \lfloor \log |a_j| \rfloor - 1$, $q = \lfloor p/(j-i) \rfloor$ and $r := p - q \cdot (j-i)$, $0 \leq r < j-i$. Then we know that

$$2^p < \left| \frac{a_i}{a_j} \right| < 2^{p+2}.$$

Taking the $(j-i)$th root we get

$$2^q < \left| \frac{a_i}{a_j} \right|^{1/(j-i)} < 2^{q+2},$$

because $q \leq p/(j-i)$ and $(p+2)/(j-i) = q + (r+2)/(j-i) \leq q+2$. But $U(A) = 2^{q+3}$, and hence we get our desired inequality.

This lemma along with (4) yields us

$$P_A < U(A) < \frac{8n}{\ln 2} R_A. \tag{5}$$

For a polynomial $A(X)$, $A(0) \neq 0$, define

$$\mathrm{PLB}(A) := \frac{1}{U(R(A))}. \tag{6}$$

Clearly, $\mathrm{PLB}(A)$ is a lower bound on the smallest positive root of $A(X)$, if it exists; otherwise, it is a lower bound on

$$\kappa(A) := \min\{\Re(z) | z \in \mathbb{C}, \Re(z) > 0, A(z) = 0\}, \tag{7}$$

i.e. the smallest real part amongst all the roots of $A(X)$ in the positive half plane. Moreover, we claim that

$$\mathrm{PLB}(A) > \frac{\kappa(A)}{16n}. \tag{8}$$

Suppose $a+\mathrm{i}b, a > 0$, is a root of $A(X)$ such that the largest real part amongst all the roots of $R(A)(X)$ is $\Re((a+\mathrm{i}b)^{-1}) = \frac{a}{a^2+|b|^2}$. Then from (5) we know that

$$\frac{1}{\mathrm{PLB}(A)} < \frac{8n}{\ln 2} \frac{a}{a^2 + |b|^2} \leq \frac{8n}{a \ln 2}.$$

Thus

$$\mathrm{PLB}(A) > a \frac{\ln 2}{8n} \geq \kappa(A) \frac{\ln 2}{8n} > \frac{\kappa(A)}{16n}.$$

The literature contains other bounds on the positive roots of a polynomial (such as [7,19,30]) that can be computed more efficiently compared to the bound by Hong. However, none of these bounds is known to satisfy an inequality similar to (4): Hong [17, Thm. 5.3] has showed the impossibility of such a criterion for the bound by Kioustelidis [19], and the same example could be used to show the impossibility for Ştefănescu's bound [30]; we can also construct examples that imply the same impossibility for the recent bounds by Akritas et al. [7]. [3] Because of these reasons, we use the bound by Hong in our analysis. We did not use a recent bound by Akritas et al. [6], which also has an $O(n^2)$ algebraic complexity and is better than Hong's bound, because the improvement is only by a constant factor, and it will not improve our complexity results. Moreover, in practice the superior quality of these two bounds compensates for their expensive computation, if not always then at least for a substantial number of cases [6].

From now on, unless mentioned otherwise, by *the continued fraction algorithm we mean the algorithm that uses the bound by Hong for computing the* PLB *function.*

Now that we have all the details of the algorithm, we face the question of its termination.

2.2. Termination

Consider the recursion tree of the procedure $\mathtt{CF}(A, M)$ initiated with $A(X) = A_{\mathrm{in}}(X) \in \mathbb{R}[X]$ and $M(X) = X$, for a square-free polynomial $A_{\mathrm{in}}(X)$. The right child of any node in this tree corresponds to the Taylor shift $X \to X + \delta$, $\delta \geq 1$, and the left child of the node corresponds to the inversion transformation $X \to (X + 1)^{-1}$. A sequence of Taylor shifts $X \to X_0 + \delta_0, X_0 \to X_1 + \delta_1, \ldots, X_{i-1} \to X_i + \delta_i$ followed by an inversion transformation $X \to (X + 1)^{-1}$ is the same as the transformation $X \to q + (1 + X)^{-1}$, where $q = \sum_{j=0}^{i} \delta_j$. Thus with each node in the recursion tree we can associate an ordinary continued fraction $[q_0, q_1, \ldots, q_m] = P_m/Q_m$, $q_i \in \mathbb{N}$, and hence the Möbius transformation $(P_m X + P_{m-1})/(Q_m X + Q_{m-1})$; note that the nodes on the rightmost path of the recursion tree are associated with the ordinary continued fraction $[q_0]$, for some $q_0 \in \mathbb{N}$, and the Möbius transformation $X + q_0$, because there are no inversion transformations along the rightmost path. Based upon the Möbius transformation $M(X) := (P_m X + P_{m-1})/(Q_m X + Q_{m-1})$ associated with a node in the recursion tree, we can further associate the polynomial

$$A_M(X) := (Q_m X + Q_{m-1})^n A_{\mathrm{in}}(M(X)).$$

Vincent had stated that if m is large enough then $A_M(X)$ will exhibit at most one sign variation. Uspensky [33, p. 298] quantified this by showing the following: Let $A_{\mathrm{in}}(X) \in \mathbb{R}[X]$ be a square-free polynomial of degree n and Δ be the smallest distance between any pair of its roots; if m is such that

$$F_{m-1} \frac{\Delta}{2} > 1 \text{ and } F_{m-1} F_m \Delta > 1 + \epsilon_n^{-1}, \tag{9}$$

[3] This is work under progress of the author jointly with Prashant Batra, Institute for Computer Technology, Hamburg University of Technology, 21071 Hamburg, Germany (batra@tuhh.de).

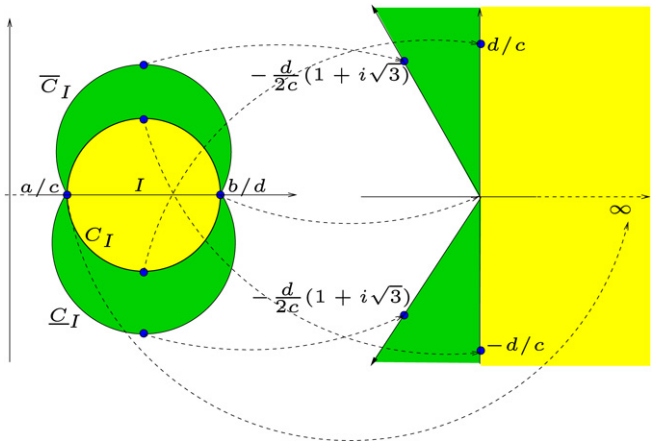

Fig. 1. The effect of $M^{-1}(z)$ on the three circles.

where F_i is the ith Fibonacci number and $\epsilon_n := (1+\frac{1}{n})^{\frac{1}{n-1}} - 1$, then $A_M(X)$ exhibits at most one sign variation. Ostrowski [25] improved and simplified Uspensky's criterion (9) to $F_m F_{m-1}\Delta \geq \sqrt{3}$. Similar criterion were derived by Alesina and Galuzzi [8, p. 246] and Yap [36, Thm. 14.5, p. 476]. However, it was Obreschkoff [24] who independently, and predating all the criteria except that by Vincent, gave the most general partial converse to the Descartes' rule of signs, which easily yields the termination criteria given below; in fact, Alesina and Galuzzi derive their criterion from this lemma; the priority amongst these criteria has been elucidated in [14, p. 17].

We next derive a termination criterion that depends on Δ_α, the shortest distance from a root α of $A(X)$ to any other root of $A(X)$. To describe this result, following [15], we associate three open discs in the complex plane with an open interval $J = (c, d)$: the disc C_J is bounded by the circle that has centre $(c+d)/2$, and radius $(d-c)/2$; the disc $\overline{C}_J$ is bounded by the circle that has centre $(c+d)/2 + \mathrm{i}(\sqrt{3}/6)(d-c)/2$, and passes through the endpoints of J; and the disc $\underline{C}_J$ is bounded by the circle that has centre $(c+d)/2 - \mathrm{i}(\sqrt{3}/6)(d-c)/2$, and passes through the endpoints of J. In addition to these three discs, again following [21], we also define the **cone**

$$\mathcal{C} := \left\{ a + \mathrm{i}b | a \leq 0 \text{ and } |b| \leq |a|\sqrt{3} \right\}.$$

We have the following key observation, which is implicit in Ostrowski's proof and is also used by Alesina and Galuzzi [8, p. 249]:

Lemma 2.4. *Let $a, b, c, d \in \mathbb{R}_{>0}$, I be an interval with unordered endpoints a/c, b/d, and define the Möbius transformation $M(z) := (az+b)/(cz+d)$. Then $M^{-1}(z)$ maps the closed region $(\mathbb{C} \cup \{\pm\infty\}) - (\overline{C}_{I_M} \cup \underline{C}_{I_M})$ bijectively on the cone $\mathcal{C}$, and maps the open disc C_{I_M} bijectively on the half plane $\Re(z) > 0$.*

For the proof verify the correspondence shown in Fig. 1.

From Lemma 2.4 and from [21, Thm. 3.9] we obtain the following result.

Theorem 2.5. *Let $A(X)$ be a square-free polynomial of degree n, and $M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}$. If α is the only simple root of $A(X)$ in the interval I_M and there are no other roots of $A(X)$ in $\overline{C}_{I_M} \cup \underline{C}_{I_M}$ then $\mathtt{Var}(A_M) = 1$.*

The above theorem corresponds to the two-circle theorem in [21]. The corresponding one-circle theorem, which again is a direct consequence of Lemma 2.4 and [21, Thm. 3.9], is the following:

Theorem 2.6. *Let $A(X)$ be a square-free polynomial of degree n, and $M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}$. If C_{I_M} does not contain any roots then $\mathtt{Var}(A_M) = 0$.*

3. The size of the recursion tree

In this section we bound the number of nodes, $\#(T)$, in the recursion tree T of the procedure described in Section 2 initiated with a square-free polynomial $A_{\mathrm{in}}(X) \in \mathbb{R}[X]$ of degree n and the Möbius transformation X.

Consider the tree T' obtained from T by pruning all leaves. Clearly, $\#(T') = \Theta(\#(T))$. Let u be a node in T' that has a right child. Furthermore, let $A_u(X)$ be the polynomial associated with u, and $A_R(X)$ be the polynomial associated with its right child. We partition the nodes u into three types:

(i) If $\mathtt{Var}(A_u) = \mathtt{Var}(A_R)$ and the number of roots of $A_u(X)$ and $A_R(X)$ in the positive half-plane are equal then u is a **type-0** node.

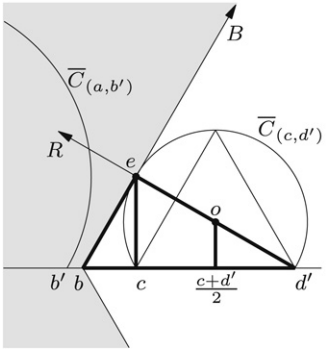

Fig. 2. Geometric proof of Lemma 3.1.

(ii) If the number of roots of $A_u(X)$ in the positive half plane is *strictly greater* than the number of roots of $A_R(X)$ in the same region, and the left child of u in T' is empty then u is a **type-1** node. This means that the Taylor shift at u shifted a set of non-real roots to the negative half plane, but the left child of u in T is a leaf; note that we may or may not loose sign variations, i.e., $\mathtt{Var}(A_u) \geq \mathtt{Var}(A_R)$, because shifting a pair of complex conjugates does not always affect the sign variations (see [14, Thm. 2.32]).

(iii) If the number of roots of $A_u(X)$ in the positive half plane is *strictly greater* than the number of roots of $A_R(X)$ in the same region, *and the left child of u in T' is not empty* then u is a **type-2** node. This means that the polynomial associated with the left child of u has at least a pair of roots in the positive half plane; thus $\mathtt{Var}(A_u) \geq \mathtt{Var}(A_R) + 2$.

Since at every type-1 and type-2 node we split the set of roots, it follows that there are at most n such nodes in T'. Thus, bounding the number of Taylor shifts in T' reduces to bounding the number of type-0 nodes in it.

Let U denote the set of leaves, U_1 the set of type-1 nodes, and U_2 the set of type-2 nodes in T'; note that the three sets are mutually disjoint. For each node $u \in U \cup U_1 \cup U_2$, let M_u denote the Möbius transformation associated with u and $I_u := I_{M_u}$. We will further associate with u a unique pair (α_u, β_u) of roots of $A_{\text{in}}(X)$.

Since the number of sign variations in the polynomial associated with every node in T' is at least two, from Theorem 2.5 it is clear that for all leaves $u \in U$ the set $\overline{C}_{I_u} \cup \underline{C}_{I_u}$ contains a pair of roots. If C_{I_u} contains a pair of complex conjugates or a pair of real roots then let (α_u, β_u) be this pair; in the former case we always choose α_u to be above the real axis, and in the latter case we always choose the roots to be a pair of neighbouring roots. However, if C_{I_u} does not contain a pair of roots then I_u must contain a real root, let α_u be this root, and $\overline{C}_{I_u} \cup \underline{C}_{I_u} - C_{I_u}$ must contain a pair of complex conjugates $(\beta_u, \overline{\beta}_u)$, where β_u is the non-real root above the real axis. Since in all the cases α_u and β_u are contained in $\overline{C}_{I_u} \cup \underline{C}_{I_u}$, it follows that

$$|I_u| > \frac{\sqrt{3}}{2}|\alpha_u - \beta_u| > \frac{|\alpha_u - \beta_u|}{2}. \tag{10}$$

For every type-1 node $u \in U_1$, we associate a pair of roots of $A_{\text{in}}(X)$ as follows: if a pair of complex conjugates was moved to the negative half plane when we were constructing the right child of a type-1 node then let (α_u, β_u) be this pair; if, however, we moved only a real root to the negative half plane then let α_u be this root and let β_u be the root nearest to it in $\overline{C}_{I_u} \cup \underline{C}_{I_u}$. Moreover, the inequality (10) between the width of the interval and the pair of complex conjugates associated with u still holds.

The pair of roots of $A_{\text{in}}(X)$ associated with a type-2 node is the pair associated with the first leaf encountered in a depth first traversal of the sub-tree rooted at the left child.

A possible problem with the above assignment is that a set of real roots $\{\alpha_{u_1}, \ldots, \alpha_{u_k}\}$, corresponding to the leaves $u_1, \ldots, u_k$ in U, may be associated with the same non-real root β, i.e., $\beta_{u_i} = \beta$, for all $i = 1, \ldots, k$. However, we will show that this can never occur in the recursion tree; this property was overlooked in [29], and will be required later to derive a tight bound on the size of the recursion tree. The proof depends upon the following result [14, Lem. 3.16].

Lemma 3.1. *Let $b < c < d$ be such that $d - c \leq 3(c - b)$. Then for all a, b' such that $a < b' \leq b$, $(\overline{C}_{(a,b')} \cup \underline{C}_{(a,b')}) \cap (\overline{C}_{(c,d)} \cup \underline{C}_{(c,d)}) = \emptyset$.*

Proof. The lemma follows if we prove that $\overline{C}_{(a,b')} \cap \overline{C}_{(c,d)} = \emptyset$; by symmetry the result holds for $\underline{C}_{(a,b')}$ and $\underline{C}_{(c,d)}$.

Let $d' := c + 3(c - b)$. Thus $d' \geq d$, and hence the result follows if we show that $\overline{C}_{(a,b')} \cap \overline{C}_{(c,d')} = \emptyset$, because $\overline{C}_{(c,d)} \subseteq \overline{C}_{(c,d')}$.

Consult Fig. 2. The ray B makes an angle of $\pi/3$ with $[b, d']$, and the ray R makes an angle of $\pi/6$ with $[b, d']$. Clearly, R passes through the center o of $\overline{C}_{(c,d')}$, and intersects B, say at e.

For all a, b', such that $a < b' \leq b$, $\overline{C}_{(a,b')}$ is contained in the shaded region. Moreover, $\overline{C}_{(c,d)} \subseteq \overline{C}_{(c,d')}$. Thus the lemma follows if we show that B is a tangent to $\overline{C}_{(c,d')}$, or, equivalently, if the segment $[d', e]$ is a diameter of $\overline{C}_{(c,d')}$.

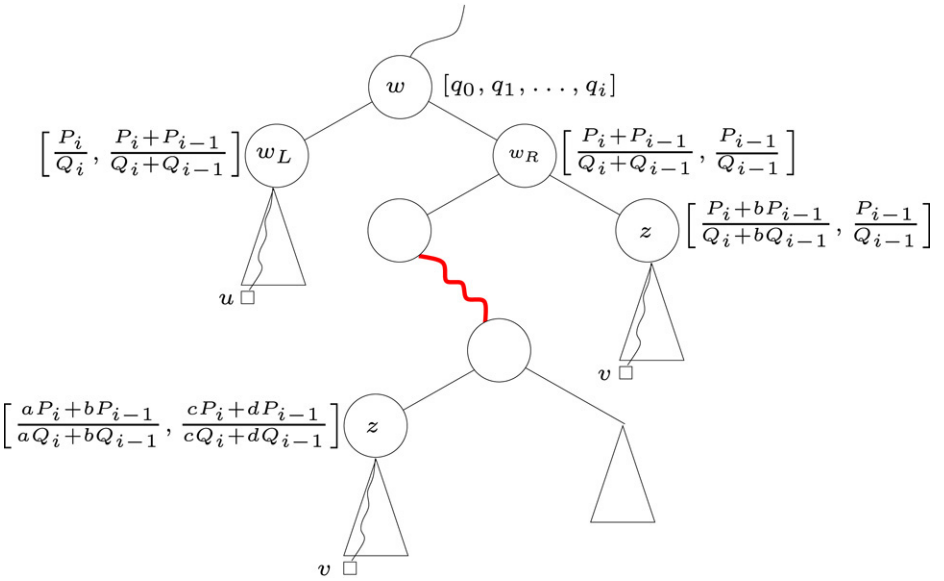

Fig. 3. Possible options for the node z such that the left endpoint of its associated interval is distinct from the right endpoint of the interval associated with w_L. The squiggly path denotes a sequence of Taylor shifts.

Consider the right-angled triangle with vertices e, b, d' ($\angle d'eb = \pi/2$). From elementary trigonometry it follows that the vertical from e to the hypotenuse $[b, d']$ meets it at c. Now if we consider the triangle with vertices e, c, d', we observe that o is the midpoint of the edge $[d', e]$, and hence this edge is a diameter of $\overline{C}_{(c,d')}$. □

Theorem 3.2. *Let u, v be two leaves in T', and I_u, I_v be the corresponding associated intervals. Then I_u and I_v do not share a common endpoint if and only if $(\overline{C}_{I_u} \cup \underline{C}_{I_u}) \cap (\overline{C}_{I_v} \cup \underline{C}_{I_v}) = \emptyset$.*

Proof. We will prove that if I_u and I_v do not share a common endpoint then $\overline{C}_{I_u} \cap \overline{C}_{I_v} = \emptyset$; the argument for $\underline{C}_{I_u}$ and $\underline{C}_{I_v}$ follows by symmetry. The proof that if I_u and I_v share a common endpoint then $\overline{C}_{I_u} \cap \overline{C}_{I_v} \neq \emptyset$ is straightforward.

Suppose the ordinary continued fraction associated with the deepest common ancestor w of u and v is $[q_0, \ldots, q_i]$, where i is an even number; if i is an odd number then the following argument still holds if we swap the endpoints of the intervals mentioned below. Further suppose that u belongs to the tree rooted at the left child, and v to the tree rooted at the right child of w; otherwise, we can exchange the roles of u and v in the following argument. Let B_1 be the value returned by the PLB function applied to the polynomial associated with w. Then the continued fraction associated with the left child, w_L, of w is $[q_0, \ldots, q_i+B_1, 1]$, and with its right child, w_R, is $[q_0, \ldots, q_i+B_1+1]$. If $P_i := P_{i-2}+(q_i+B_1)P_{i-1}$ and $Q_i := Q_{i-2}+(q_i+B_1)Q_{i-1}$ then the interval associated with w_L is $I_L := [P_i/Q_i, (P_i+P_{i-1})/(Q_i+Q_{i-1})]$ and with w_R is $[(P_i+P_{i-1})/(Q_i+Q_{i-1}), P_{i-1}/Q_{i-1}]$. This scenario has been illustrated in Fig. 3.

If the left endpoint of I_v is not equal to $(P_i + P_{i-1})/(Q_i + Q_{i-1})$ then in the tree rooted at w_R there must be a node z with the least depth such that the interval I_z associated with z has a left endpoint greater than $(P_i + P_{i-1})/(Q_i + Q_{i-1})$, and such that v is equal to, or a descendant of this node, i.e., $I_v \subseteq I_z$. There are two possibilities for z: first, it is the right child of w_R; and second, it is either the left child of w_R or a node obtained from it in a sequence (possibly empty) of Taylor shifts followed by an inversion transformation. In both cases we claim that

$$I_z = \left[\frac{aP_i + bP_{i-1}}{aQ_i + bQ_{i-1}}, \frac{cP_i + dP_{i-1}}{cQ_i + dQ_{i-1}}\right],$$

where $a, b, c, d \in \mathbb{N}$ are such that $a < b, c < d$, and $ad - bc = 1$; the case $c = 0$ corresponds to the right child of w_R. We show the claim for the second case; the proof for the first case can be shown similarly. Using the matrix representation (see e.g., [36, p. 462]) of a Möbius transformation it follows that the transformation associated with z in the second case is

$$\begin{bmatrix} P_i + P_{i-1} & P_{i-1} \\ Q_i + Q_{i-1} & Q_{i-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \delta_1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \delta_2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ X \end{bmatrix}, \tag{11}$$

where $\delta_1 \geq 0$ is the amount of Taylor shift done at w_R, and $\delta_2 \geq 0$, is the *total amount* of Taylor shift done along the squiggly path shown in Fig. 3. From this it follows that $a = 2 + \delta_2$, $b = 1 + (1 + \delta_1)a$, $c = 1 + \delta_2$, $d = 1 + (1 + \delta_1)c$, and $ad - bc = 1$. Note that (11) does not represent the transformation associated with the left child of w_R; nevertheless, from the same argument it is clear that the associated transformation satisfies $a = 1$, $b = 2 + \delta_1$, $c = 1$, and $d = 1 + \delta_1$.

Now we show that $\overline{C}_{I_z} \cap \overline{C}_{I_L} = \emptyset$. From Lemma 3.1 this follows if

$$|I_z| < 3\left|\frac{aP_i + bP_{i-1}}{aQ_i + bQ_{i-1}} - \frac{P_i + P_{i-1}}{Q_i + Q_{i-1}}\right|,$$

i.e., if

$$Q_i + Q_{i-1} < 3(b-a)(cQ_i + dQ_{i-1}).$$

This certainly holds if $c \geq 1$. If, however, $c = 0$ then $a = d = 1$, because $ad - bc = 1$, and $b > 1$, because otherwise the left endpoint of J is $(P_i + P_{i-1})/(Q_i + Q_{i-1})$. But even in this situation $\overline{C}_{I_z} \cap \overline{C}_{I_L} = \emptyset$, because

$$|I_L| = \left|\frac{P_i}{Q_i} - \frac{P_i + P_{i-1}}{Q_i + Q_{i-1}}\right| < 3\left|\frac{P_i + P_{i-1}}{Q_i + Q_{i-1}} - \frac{P_i + bP_{i-1}}{Q_i + bQ_{i-1}}\right|,$$

since for $b > 1$, $Q_i + bQ_{i-1} < 3(b-1)Q_i$.

If the left endpoint of I_v is $(P_i + P_{i-1})/(Q_i + Q_{i-1})$ then by assumption the right endpoint of I_u has to be smaller than this. Thus in the tree rooted at w_L there must be a node z whose associated interval I_z has a right endpoint smaller than $(P_i + P_{i-1})/(Q_i + Q_{i-1})$ and such that v is equal to, or a descendant of this node, i.e., $I_v \subseteq J$. Moreover, we can again show that

$$I_z = \left[\frac{aP_i + bP_{i-1}}{aQ_i + bQ_{i-1}}, \frac{cP_i + dP_{i-1}}{cQ_i + dQ_{i-1}}\right],$$

where $a, b, c, d \in \mathbb{N}$ are such that $a > b, c > d, ad - bc = 1$, and $a, c \geq 1$. Again, from Lemma 3.1 we get $\overline{C}_{I_z} \cap \overline{C}_{I_L} = \emptyset$, because

$$|I_z| \leq 3\left|\frac{cP_i + dP_{i-1}}{cQ_i + dQ_{i-1}} - \frac{P_i + P_{i-1}}{Q_i + Q_{i-1}}\right|. \quad \square$$

We next bound the number of nodes in the path terminating at the leaf u by bounding the number of inversion transformations $X \to 1/(X+1)$ and Taylor shifts $X \to X + b, b \geq 1$.

3.1. Bounding the inversion transformations

For a given leaf $u \in U$, let $[q_0, , \ldots, q_m] = P_m/Q_m$ be the associated continued fraction. Then from (10) it follows that $|\alpha_u - \beta_u| < 2(Q_m Q_{m-1})^{-1}$. But from (1) we know that Q_i is greater than the $(i+1)$th Fibonacci number; this implies that $Q_i \geq \phi^i$, where $\phi = (1 + \sqrt{5})/2$. Thus $\phi^{2m-1} \leq 2|\alpha_u - \beta_u|^{-1}$ and hence

$$m \leq \frac{1}{2}(1 + \log_\phi 2 - \log_\phi |\alpha_u - \beta_u|). \tag{12}$$

This gives us an upper bound on the number of inversion transformations for one leaf u; summing it for all leaves gives us the following bound on the total number of inversion transformations in T'

$$\sum_{u \in U} \frac{1}{2}(1 + \log_\phi 2 - \log_\phi |\alpha_u - \beta_u|) \leq 2n + \sum_{u \in U} \log_\phi (|\alpha_u - \beta_u|)^{-1}. \tag{13}$$

This bound is also derived in [31, Thm. 8], and improves upon Akritas' bound [5] by a factor of n.

3.2. Bounding the Taylor shifts

The purpose of the Taylor shifts in the procedure $\mathtt{CF}(A, M)$ was to compute the floor of the smallest positive root of a polynomial. Using property (8) of the $\mathrm{PLB}(A)$ function (defined in (6)) we will bound the number of Taylor shifts required to compute the floor of the smallest positive root of some polynomial $B(X) \in \mathbb{R}[X]$.

We introduce the following notation for convenience: for any $x \in \mathbb{R}_{\geq 0}$ let

$$\mathrm{logm}(x) := \log \max(1, x).$$

Lemma 3.3. *Let $B_1(X) \in \mathbb{R}[X]$. For $i > 1$ recursively define*

$$B_i(X) := B_{i-1}(X + \delta_{i-1})$$

where

$$\delta_{i-1} := \begin{cases} \mathrm{PLB}(B_{i-1}) + 1 & \text{if } \mathrm{PLB}(B_{i-1}) \geq 1 \\ 1 & \text{otherwise.} \end{cases}$$

Let $\alpha_1 := \kappa(B_1(X))$ (see (7)), and a_1 be any point in $[0, \alpha_1]$. Recursively define $\alpha_i := \alpha_{i-1} - \delta_{i-1}$ and $a_i := a_{i-1} - \delta_{i-1}$. Then $a_i \leq 1$ if $i = \Omega(n + n\mathrm{logm}\, a_1)$.

Proof. Let $b_i := \mathrm{PLB}(B_i)$. Then from (8) we know that $b_i > \alpha_i/16n$, because $\alpha_i = \kappa(B_i)$. Let j be the index such that $\alpha_i > 16n$ for $i < j$. Then for $i < j$ we know that $b_i > 1$. Thus

$$a_i = a_{i-1} - b_{i-1} - 1 < a_{i-1} - b_{i-1} < a_{i-1} - \frac{\alpha_{i-1}}{16n} \leq a_{i-1}\left(1 - \frac{1}{16n}\right),$$

because $a_i \leq \alpha_i$, for all i. Thus recursively we obtain $a_i < a_1(1 - \frac{1}{16n})^{i-1}$. So $a_j \leq 16n$ if $j \geq 2 + (\log(16n) - \log(16n - 1))^{-1}\log a_1$. For $i \geq j$ we know that $\alpha_i \leq 16n$, because α_i is monotonically decreasing. Thus if $i > j$ is such that $i - j \geq 16n$ then $a_i \leq 1$. Combining this lower bound on $i - j$ with the lower bound on j along with the observation that $(\log(16n) - \log(16n - 1))^{-1} = \Theta(n)$ we get the result of the lemma. □

Based upon the above lemma we will bound the number of Taylor shifts from the root of T' to the leaf u, with the associated continued fraction $[q_0, \ldots, q_m]$, by bounding the number of Taylor shifts that compose each q_i, $i = 0, \ldots, m$. Recall from the beginning of this section that for every leaf $u \in U$ we had associated a pair (α_u, β_u) of roots of $A_{\mathrm{in}}(X)$; let $M_u(X)$ be the Möbius transformation and I_u the interval associated with u. Further define the following quantities:

Definition 3.4. For $0 \leq i \leq m$ let

(i) $M_i(X) := [q_0, \ldots, q_i, 1 + X] = \frac{P_iX + P_{i-1} + P_i}{Q_iX + Q_{i-1} + Q_i}$, and
(ii) $A_i(X) := (Q_iX + Q_{i-1} + Q_i)^n A_{\mathrm{in}}(M_i(X))$, i.e., the polynomial obtained by performing the ith inversion transformation and on which we will perform a Taylor shift by the amount q_{i+1}.

By its definition I_{M_i}, i.e. the interval associated with $M_i(X)$, for $0 \leq i \leq m$, contains I_u and hence it follows from (10) that for $0 \leq i \leq m$

$$(Q_iQ_{i-1})^{-1} \geq \frac{|\alpha_u - \beta_u|}{2}. \tag{14}$$

We now bound the number of Taylor shifts required by the algorithm to compute q_{i+1}. Suppose the scenario is as shown in Fig. 4, i.e., while computing q_{i+1} we have one type-2 node w_1 with the associated continued fraction $[q_0, \ldots, q_i, q_{w_1}]$ and one type-1 node u_1 with the associated continued fraction $[q_0, \ldots, q_i, q_{u_1}] = P_{u_1}/Q_{u_1}$; the squiggly paths represent the type-0 nodes in between.

As illustrated in Fig. 4, the number of Taylor shifts needed to compute q_{i+1} can be broken into three parts: first, the number of Taylor shifts required to reach w_1; second, the number of Taylor shifts required to go from w_1 to u_1; and third, the number of Taylor shifts needed from u_1. From Lemma 3.3 we know that the first quantity is $O(n \log q_{w_1})$, the second quantity is $O(n \log(q_{u_1} - q_{w_1})) = O(n \log q_{u_1})$, and the third is $O(n \log(q_{i+1} - q_{w_1} - q_{u_1})) = O(n \log q_{i+1})$.

Consider the bound $O(n \log q_{w_1})$. Recall that with every type-2 node there is a unique pair of roots associated; suppose the pair (α_v, β_v) associated with w_1 corresponds to the leaf $v \in U$. As shown in Fig. 4, it is clear that the partial denominator q_{w_1} is smaller than the $(i + 1)$th partial denominator q_v appearing in the continued fraction associated with v. Thus $O(n \log q_{w_1}) = O(n \log q_v)$. Also recall that with every type-1 node we had associated a unique pair of roots; say $(\alpha_{u_1}, \beta_{u_1})$ is the pair associated with u_1. We know that the width of the interval associated with u_1 satisfies $2(Q_iQ_{u_1})^{-1} > |\alpha_{u_1} - \beta_{u_1}|$. Since $Q_{u_1} > q_{u_1}$ we get $O(n \log q_{u_1}) = O(n \log |\alpha_{u_1} - \beta_{u_1}|^{-1})$.

The above argument holds in general, i.e., while constructing $A_{i+1}(X)$ from $A_i(X)$ suppose $u_1, u_2, \ldots, u_k$, $k \geq 0$, are the type-1 nodes, $w_1, w_2, \ldots, w_\ell$, $\ell \geq 0$, are the type-2 nodes, and all the remaining nodes are of type-0. Let $(\alpha_{u_j}, \beta_{u_j})$ represent the pair associated with the type-1 node u_j, and $(\alpha_{v_j}, \beta_{v_j})$, $v_j \in U$, represent the pair associated with the type-2 node w_j; further define q_{v_j} as the $(i+1)$th partial denominator appearing in the continued fraction associated with v_j. Then the total number of Taylor shifts needed to compute $A_{i+1}(X)$ from $A_i(X)$ is bounded by

$$O\left(n \sum_{j=1}^{k} \log |\alpha_{u_j} - \beta_{u_j}|^{-1} + n \sum_{j=1}^{\ell} \log q_{v_j} + n \log q_{i+1}\right). \tag{15}$$

If we sum this bound for $i = 1, \ldots, m$, and then for all $u \in U$ we get an upper bound on the total number of Taylor shifts in the tree T'. We consider this summation in three separate parts corresponding to the three terms appearing in (15).

(i) The summation over $i = 1, \ldots, m$, and over all $u \in U$, of the first summation term in (15) amounts to summing over all the type-1 nodes appearing in the tree, and hence is bounded by $\sum_{u' \in U_1} O(n \log |\alpha_{u'} - \beta_{u'}|^{-1})$.
(ii) Let us consider the summation $\sum_{i=1}^{m} \log q_i$; we will later consider the bound for q_0 for all $u \in U$, i.e., a bound on the length of the rightmost path. We express this summation in terms of the distance $|\alpha_u - \beta_u|$. We know that $Q_m = q_mQ_{m-1} + Q_{m-2}$; thus $Q_m \geq q_mQ_{m-1}$, and recursively we get that $Q_m \geq \prod_{j=1}^{m} q_j$. Along with (14) we have

$$\sum_{i=1}^{m} \log q_i = O(\log |\alpha_u - \beta_u|^{-1}). \tag{16}$$

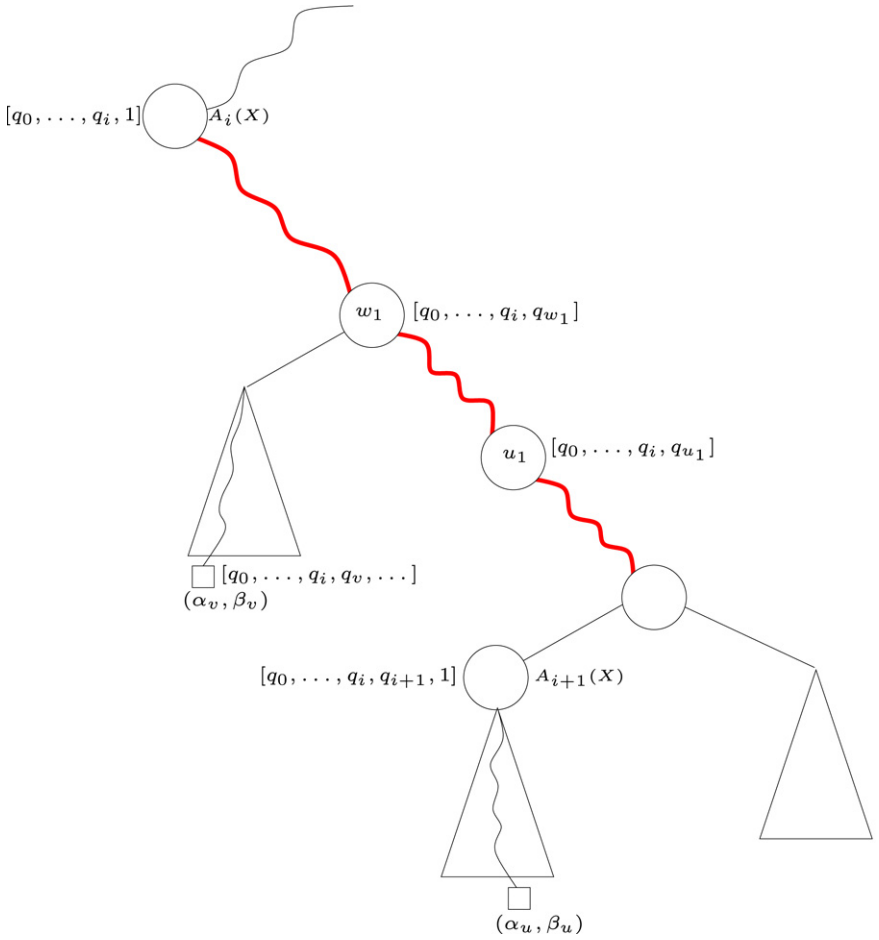

Fig. 4. Taylor shifts between the ith and $(i+1)$th inversion transformations.

(iii) Consider the second summation in (15). Each term in the summation is the $(i+1)$th partial denominator in the continued fraction associated with the leaf v_j. But if we consider the bound in (16) for the leaf v_j then we account for the term $\log q_{v_j}$. Thus the summation of (16) over all $u \in U$ bounds the number of Taylor shifts associated with all the type-2 nodes in the tree T'.

We have accounted for all the shifts in the tree, except for the shifts along the rightmost path in T'. Let q_0^u be the 0th partial numerator in the continued fraction associated with leaf u. Then from Lemma 3.3 it follows that the length of the rightmost path is $O(n \sum_{u \in U} \log q_0^u)$. Since each q_0^u is smaller than the largest absolute value amongst all the roots of $A_{\text{in}}(X)$, which instead is bounded by $\mathrm{M}(A_{\text{in}})/|\text{lead}(A_{\text{in}})|$, where $\mathrm{M}(A_{\text{in}})$ is the Mahler measure of $A_{\text{in}}(X)$ and $\text{lead}(A_{\text{in}})$ is its leading coefficient [36, Sec. 6.6, Sec. 4.5], we have the following result.

Lemma 3.5. *The total number of Taylor shifts in the tree T' is bounded by*

$$O\left(n \sum_{u \in U \cup U_1} \log |\alpha_u - \beta_u|^{-1} + n^2 \log \frac{\mathrm{M}(A_{in})}{|\text{lead}(A_{in})|}\right).$$

Asymptotically this bound dominates the bound in (13) on the total number of inversion transformations in the tree T' and hence is also a bound on $\#(T')$, the size of the tree T'. Moreover, recall from the beginning of this section that the number of nodes in the recursion tree T of the continued fraction algorithm is $\Theta(\#(T'))$, so the above bound applies to $\#(T)$ as well.

3.3. Worst case size of the tree

In order to derive a worst case bound on the size of the tree T, from the bound given in Lemma 3.5, we need to derive an upper bound on $\sum_{u \in U \cup U_1} \log |\alpha_u - \beta_u|^{-1}$. For this purpose we resort to the Davenport–Mahler bound:

Proposition 3.6. *Let $A(X) = a_n \prod_{i=1}^{n}(X - \alpha_i)$ be a square-free complex polynomial of degree n. Let $G = (V, E)$ be a directed graph whose vertices $\{v_1, \ldots, v_k\}$ are a subset of the roots of $A(X)$ such that*

(i) *If $(v_i, v_j) \in E$ then $|v_i| \leq |v_j|$.*
(ii) *G is acyclic.*
(iii) *The in-degree of any node is at most* 1.

If exactly m of the nodes have in-degree 1, *then*

$$\prod_{(v_i,v_j)\in E} |v_i - v_j| \geq \sqrt{|\mathrm{discr}(A)|} \cdot \mathrm{M}(A)^{-(n-1)} \cdot (n/\sqrt{3})^{-m} \cdot n^{-n/2}.$$

See [9, Prop. 10.23] for a proof. A special case of this proposition is the following:

Remark 3.7. Let $A(X)$ be square-free integer polynomial of degree n and integer coefficients of bit-length L. Let $\mathrm{sep}(A)$ be the minimum distance between two distinct roots of $A(X)$. Then

$$\mathrm{sep}(A) \geq 2^{-nL}(n+1)^{-n} \cdot (n/\sqrt{3}).$$

The bound is not the sharpest (cf. [27]), but is sufficient for asymptotic analysis.

Consider the graph G whose edge set consists of the pairs (α_u, β_u), $u \in U \cup U_1$. We will show that the edges in G can be reordered to satisfy the conditions of Proposition 3.6. First of all, for any $u \in U \cup U_1$ we can reorder the pair (α_u, β_u) to ensure that $|\alpha_u| \leq |\beta_u|$ without affecting the summation in Lemma 3.5. However, we might have a situation where two nodes $u, u' \in U \cup U_1$ with the associated pairs of roots (α_u, β_u) and $(\alpha_{u'}, \beta_{u'})$ (resp.) are such that $\beta_u = \beta_{u'}$; from Theorem 3.2 it is clear that this can occur if and only if the intervals associated with u and u' share a common endpoint; moreover, in this case, β_u is a non-real root. In this situation we can always re-assign $\beta_{u'} := \overline{\beta_u}$. Now we are sure that all the vertices in G have in-degree at most one. Applying Proposition 3.6 to G we obtain

$$\sum_{u\in U\cup U_1} \log |\alpha_u - \beta_u|^{-1} = O(n \log \mathrm{M}(A_{\mathrm{in}}) + n \log n - \log|\mathrm{discr}(A_{\mathrm{in}})|), \tag{17}$$

where $\mathrm{M}(A_{\mathrm{in}})$ is the Mahler measure of $A_{\mathrm{in}}(X)$ and $\mathrm{discr}(A_{\mathrm{in}})$ is its discriminant (see [36, Sec. 6.6, Sec. 4.5], [23, Sec. 1.5, Sec. 2.1]). Based upon this bound, we have the following result.

Theorem 3.8. *The number of nodes in the recursion tree of the procedure CF applied to a square-free polynomial $A_{in}(X) \in \mathbb{R}[X]$ of degree n is*

$$O(n^2 \log \mathrm{M}(A_{in}) + n^2 \log n - n \log|\mathrm{discr}(A_{in})| - n^2 \log |\mathrm{lead}(A_{in})|).$$

Proof. The result follows easily if we apply the bound in (17) to the bound in Lemma 3.5. □

We next give a specialization of the above theorem for integer polynomials.

Corollary 3.9. *The number of nodes in the recursion tree of the procedure CF applied to a square-free polynomial $A_{in}(X)$ of degree n with integer coefficients of bit-length L is $\widetilde{O}(n^2L)$.*

Proof. From Landau's inequality (e.g., [36, Lem. 4.14(i)]) and the estimate $\|A_{\mathrm{in}}\|_2 \leq \sqrt{n+1}\|A_{\mathrm{in}}\|_\infty$ we get that

$$\mathrm{M}(A_{\mathrm{in}}) \leq \|A_{\mathrm{in}}\|_2 \leq \sqrt{n+1}\|A_{\mathrm{in}}\|_\infty < 2^L\sqrt{n+1}.$$

Moreover, $|\mathrm{discr}(A_{\mathrm{in}})|, |\mathrm{lead}(A_{\mathrm{in}})| \geq 1$, because $A_{\mathrm{in}}(X)$ is square-free and its coefficients are integers. □

How good is this bound? The answer depends upon the tightness of the bounds derived on the number of inversion transformations and Taylor shifts. The bound derived on the former, in (13), is perhaps the best one can expect, considering that the same bound holds for real root isolation using Sturm's method [12,13] and for Collins and Akritas' algorithm [15, 16]. The best possible scenario for the number of Taylor shifts is based upon the ideal PLB function, which we recall is a function that can determine whether a polynomial has positive real roots, and if there are such roots then returns the floor of the smallest positive root of the polynomial.

Lemma 3.10. *For a square-free polynomial of degree n and integer coefficients of bit-length L, the number of Taylor shifts in the recursion tree of the procedure CF using the ideal* PLB *function is $\widetilde{O}(nL)$.*

Proof. Recall the definition of the polynomials $A_i(X)$ from Definition 3.4. Notice that in the process of constructing $A_{i+1}(X)$ from $A_i(X)$, each Taylor shift using the bound provided by the ideal PLB function ensures that there is a set of real roots of $A_i(X)$ in the unit interval. Thus each Taylor shift on $A_i(X)$ either partitions its positive real roots, or is immediately followed by an inversion transformation if all the real roots are contained in a unit interval. Thus the total number of Taylor shifts in the recursion tree is less than sum of the degree n and the number of inversion transformations in the tree. In the worst case this sum is bounded by $\widetilde{O}(nL)$. □

Remark 3.11. Combining the arguments presented in Section 3.2 and the arguments in [28, Chap. 3], we can show an $\widetilde{O}(n^3L)$ bound on the recursion tree size of procedure CF, where we use Zassenhaus' bound [36, Lem. 6.5, p. 147] for computing the PLB function. If, however, in the arguments in [28, Chap. 3] we use Roth's theorem [26] instead of Liouville's inequality then the bound can be reduced by a factor of n.

4. The Bit-complexity

In this section we derive the bit-complexity of the continued fraction algorithm for a square-free integer polynomial $A_{\text{in}}(X)$ such that $\|A_{\text{in}}\|_\infty < 2^L$. To do this we will bound the worst case complexity at any node in the recursion tree, and multiply it with the bound in Corollary 3.9.

Recall from the beginning of Section 3 the definitions of the set U, and of the pair (α_u, β_u) for any $u \in U$. Let $[q_0, \ldots, q_{m+1}] = P_{m+1}/Q_{m+1}$, $A_i(X)$ be defined as in Definition 3.4, and $\|A_i\|_\infty < 2^{L_i}$.

To construct $A_{i+1}(X)$ from $A_i(X)$ we need to construct a sequence of polynomials $B_j(X)$, $1 \le j \le \ell$, such that $B_1(X) := A_i(X)$; for $j > 1$, $B_j(X) := B_{j-1}(X + \delta_{j-1})$, where δ_j is defined as in Lemma 3.3; and $A_{i+1}(X) := (X+1)^n B_\ell(\delta_\ell + 1/(X+1))$. Moreover, $q_{i+1} = \sum_{j=1}^{\ell} \delta_j$. The two most important operations in computing $B_j(X)$ from $B_{j-1}(X)$ are computing $\text{PLB}(B_{j-1})$ and the Taylor shift by δ_j. We only focus on the latter operation, because if we use the classical Taylor shifts then its complexity dominates the cost of computing $\text{PLB}(B_{j-1})$.

In order to compute $B_j(X)$ from $B_{j-1}(X)$, we first perform a Taylor shift by $\text{PLB}(B_{j-1})$, and then by one. If we use classical Taylor shifts [20,35] then we require $O(n^2)$ additions and multiplications on integers of bit-length $O(b_{j-1} + n \log \delta_{j-1})$, where b_{j-1} is the bit-length of the coefficients of $B_{j-1}(X)$. Recall from Section 2.1 that the value returned by $\text{PLB}(B_{j-1})$ is a power of two, and hence all the multiplications in the implementation of Taylor shift are replaced by simple bit-shift operations. Thus the cost of computing $B_{j-1}(X + \text{PLB}(B_{j-1}))$ is bounded by $O(n^2(b_{j-1} + n \log \delta_{j-1}))$. Recursively, we can show that $b_{j-1} = O(L_i + n(\log \delta_1 + \cdots + \log \delta_{j-2}))$. In particular, the cost of computing $A_{i+1}(X)$ from $B_\ell(X)$ is bounded by

$$O\left(n^2\left(L_i + n\sum_{j=1}^{\ell} \log \delta_j\right)\right) = O(n^2(L_i + n\ell \log q_{i+1})).$$

Let us write ℓ_i to denote ℓ. Then the worst case complexity of computing $A_{i+1}(X)$ is bounded by

$$O\left(n^2\left(L + n\sum_{i=1}^{m} \ell_i \log q_{i+1}\right)\right) = O\left(n^2\left(L + n\left(\sum_{i=1}^{m} \ell_i\right)\left(\sum_{i=1}^{m} \log q_{i+1}\right)\right)\right).$$

From (16) and Remark 3.7, we know that $\sum_{i=1}^{m} \log q_{i+1} = \widetilde{O}(nL)$; and from Corollary 3.9, that $\sum_{i=1}^{m} \ell_i = \widetilde{O}(n^2 L)$. Thus the worst case complexity at a node is bounded by $\widetilde{O}(n^6 L^2)$; at the leaves the worst case complexity is $\widetilde{O}(n^7 L^2)$, because q_{m+1} does not satisfy (16), but from from [28, p. 158] we know that $b_{m+1} = \widetilde{O}(n^2 L)$; however, the number of leaves is at most $O(n)$, but the number of remaining nodes in the recursion tree is $O(n^2 L)$. Thus we have the following result.

Theorem 4.1. *Let $A(X)$ be a square-free integer polynomial of degree n with integer coefficients of bit-length L. Then the bit-complexity of isolating all the real roots of $A(X)$ using the procedure CF, where we use classical Taylor shift, is $\widetilde{O}(n^8 L^3)$.*

Note that we cannot use asymptotically fast Taylor shifts [35, Thm. 2.4] to improve this bound, because then its complexity is dominated by the cost of computing the bound by Hong.

5. Continued fraction algorithm with scaling

The results in the previous section show that even though shifting by PLB in step five of the procedure $\texttt{CF}(A, M)$ brings down the worst case running time to polynomial, it does not yield an impressive bound. Another way to perform a Taylor shit by an amount $b \ge 1$ is to scale the variable in the polynomial by b and then do a Taylor shift by one. So we can replace line five of procedure $\texttt{CF}(A, M)$ with the following:

> 5. **If** $b \ge 1$ **then**
> $A(X) := A(b(X+1)), M(X) := M(b(X+1))$.

Let $\texttt{CFS}(A, M)$ be the modified procedure; it was first proposed in [3].

To analyse this modification, we have to understand the nature of the Möbius transformations associated with each node in the recursion tree of the formulation. As was the case earlier, the transformation associated with the root of the tree is X. Now suppose we perform a scaling by some amount b_0 followed by a Taylor shift by one. Then the associated transformation is $b_0 + b_0 X$. If we follow this by a scaling by b_1 and a Taylor shift by one then the associated transformation is $b_0 + b_0 b_1 + b_0 b_1 X$. If the next transformation is an inversion transformation then the associated transformation is $b_0 + b_0 b_1 + \frac{b_0 b_1}{1+X}$. Thus we see that the continued fraction associated with a node of the tree is of the form

$$\left[q_0, \frac{p_1}{q_1}, \frac{p_2}{q_2}, \ldots, \frac{p_m}{q_m}\right],$$

i.e., the *continued fraction is no longer guaranteed to be an ordinary continued fraction*. Moreover, an important property of this continued fraction, which can be deduced from the explanation above, is that

$$q_i \ge p_{i+1}. \tag{18}$$

The termination criteria, namely Theorems 2.5 and 2.6, still hold.

Let T be the recursion tree of the procedure CFS(A_{in}, M), where $A_{\text{in}}(X) \in \mathbb{R}[X]$ is a square-free polynomial of degree n and $M(X)$ is the identity Möbius transformation. Our aim is to bound the size $\#(T)$ of T. We proceed similar to Section 3. In particular, let T' denote the tree obtained by pruning all the leaves of T. We again characterize the set of nodes of T' as type-0, type-1 and type-2 nodes; let U denote the set of leaves, U_1 the set of type-1 nodes, and U_2 the set of type-2 nodes. For each $u \in U$ we can associate a pair of roots (α_u, β_u) of $A_{\text{in}}(X)$ as was done earlier; the argument below does not require associating any pair of roots with type-1 and type-2 nodes. However, for the same reasons given earlier, we have to prove that Theorem 3.2 still holds for T'.

Theorem 5.1. *Let u, v be two leaves in T', and I_u, I_v be the corresponding associated intervals. Then I_u and I_v do not share a common endpoint if and only if $(\overline{C}_{I_u} \cup \underline{C}_{I_u}) \cap (\overline{C}_{I_v} \cup \underline{C}_{I_v}) = \emptyset$.*

Proof. We proceed similarly to Theorem 3.2. Let w be the deepest common ancestor of u and v, $[q_0, \frac{p_1}{q_1}, \ldots, \frac{p_i}{q_i}, p_{i+1}]$ be the continued fraction associated with w, where i is an even number, and B_1 be the value returned by the PLB function applied to the polynomial associated with w. Define $P'_{i-1} := B_1 p_{i+1} P_{i-1}$, $Q'_{i-1} := B_1 p_{i+1} Q_{i-1}$, $P_i := p_i P_{i-2} + q_i P_{i-1} + P'_{i-1}$, and $Q_i := p_i Q_{i-2} + q_i Q_{i-1} + Q'_{i-1}$. Then the interval I_L associated with the left child w_L of w is $[P_i/Q_i, (P_i + P'_{i-1})/(Q_i + Q'_{i-1})]$ and with the right child w_R of w is $[(P_i + P'_{i-1})/(Q_i + Q'_{i-1}), P'_{i-1}/Q'_{i-1}]$. The scenario is the same as in Fig. 3 with P_{i-1}, Q_{i-1} replaced by P'_{i-1}, Q'_{i-1} respectively.

Again, if the left endpoint of I_v is not equal to $(P_i + P'_{i-1})/(Q_i + Q'_{i-1})$ then in the tree rooted at w_R there must be a node z with the least depth such that the interval I_z associated with z has a left endpoint greater than $(P_i + P'_{i-1})/(Q_i + Q'_{i-1})$, and such that v is equal to, or a descendant of this node, i.e., $I_v \subseteq I_z$. There are two cases to consider for z: first, it is the right child of w_R; and second, it is either the left child of w_R or a node obtained from it in a sequence (possibly empty) of Taylor shifts and scalings, followed by an inversion transformation. In both cases we claim that

$$I_z = \left[\frac{aP_i + bP_{i-1}}{aQ_i + bQ_{i-1}}, \frac{cP_i + dP_{i-1}}{cQ_i + dQ_{i-1}}\right],$$

where $a, b, c, d \in \mathbb{N}$ are such that $a < b$, $c < d$, and $ad - bc \geq 1$ is the total amount of scaling done on the path to z.

We have to consider sub-cases of the second case. First assume that there was no scaling at w_R, i.e., the PLB function applied to the polynomial associated with w_R returned zero; in this sub-case z cannot be the left child of w_R. Then the Möbius transformation associated with z is

$$\begin{bmatrix} P_i + P'_{i-1} & P'_{i-1} \\ Q_i + Q'_{i-1} & Q'_{i-1} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \delta_1 & \delta_1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 0 \\ \delta_k & \delta_k \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ X \end{bmatrix}, \tag{19}$$

where $\delta_1, \ldots, \delta_k \geq 1$ are the scalings done along the squiggly path shown in Fig. 3; note that $\delta_j = 1$ just means a Taylor shift by one. In this case $a = 1 + \sum_{j=1}^{k} \delta_1 \ldots \delta_j + \delta_1 \ldots \delta_k$, $b = 1 + a$, $c = 1 + \sum_{j=1}^{k} \delta_1 \ldots \delta_j$, $d = 1 + c$ and $ad - bc = \delta_1 \ldots \delta_k$. However, (19) does not represent the transformation when there are no scalings and Taylor shifts, i.e., the sub-case when z is the left child of the left child of w_R. That transformation is

$$\begin{bmatrix} P_i + P'_{i-1} & P'_{i-1} \\ Q_i + Q'_{i-1} & Q'_{i-1} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ X \end{bmatrix};$$

in this case $a = 2, b = 3, c = 1$, and $d = 2$. If our assumption was wrong, i.e., there was a non-zero scaling $\delta \geq 1$ done at w_R then the transformation associated with its left child, which is our z now, satisfies $a = 1, b = 2\delta + 1, c = 1$, and $d = \delta + 1$. In all the three sub-cases, it can be easily shown that

$$|I_z| < 3 \left| \frac{aP_i + bP'_{i-1}}{aQ_i + bQ'_{i-1}} - \frac{P_i + P'_{i-1}}{Q_i + Q'_{i-1}} \right|,$$

i.e., $|ad - bc|(Q_i + Q'_{i-1}) < 3(b - a)(cQ_i + dQ'_{i-1})$. This argument works if $c \neq 0$. If that is not the case, which is the situation when z is the right child of w_R, then, if $\delta \geq 1$ is the value returned by PLB function at w_R, the transformation associated with its right child satisfies $a = 1, b = 2(\delta + 1), c = 0$, and $d = \delta$. In this case we can show

$$|I_L| = \left| \frac{P_i}{Q_i} - \frac{P_i + P'_{i-1}}{Q_i + Q'_{i-1}} \right| < 3 \left| \frac{P_i + P'_{i-1}}{Q_i + Q'_{i-1}} - \frac{P_i + bP'_{i-1}}{Q_i + bQ'_{i-1}} \right|,$$

i.e., $Q_i + bQ'_{i-1} < 3(b - 1)Q_i$.

The argument when the left endpoint of I_v is $(P_i + P'_{i-1})/(Q_i + Q'_{i-1})$ can be done similarly. □

5.1. Bounding the inversion transformations

The argument in Section 3.1 breaks down now, because the width of the interval associated with the transformation $(P_mX + P_{m-1})/(Q_mX + Q_{m-1})$ is $(p_1p_2 \dots p_m)/(Q_mQ_{m-1})$. To have a similar argument we need to derive an upper bound on $(p_1p_2 \dots p_m)/(Q_mQ_{m-1})$. We can show a trivial upper bound of one, but it is not tight enough for carrying our argument. However,

$$Q_m = p_mQ_{m-2} + q_mQ_{m-1} \geq q_mQ_{m-1} \geq \prod_{i=1}^{m} q_i \geq \prod_{i=1}^{m} p_{i+1}, \tag{20}$$

where the last inequality follows from (18).

Now we bound the number of inversion transformations needed along a path to a leaf $u \in U$ with the associated continued fraction $[q_0, \frac{p_1}{q_1}, \dots, \frac{p_m}{q_m}] = P_m/Q_m$. From (10) it follows that

$$|\alpha_u - \beta_u| < 2\left|\frac{P_m}{Q_m} - \frac{P_{m-1}}{Q_{m-1}}\right| = 2\frac{p_1p_2 \dots p_m}{Q_mQ_{m-1}} \leq \frac{2}{Q_m}, \tag{21}$$

where the last step follows from (20). Since $Q_m \geq \phi^m$, where $\phi = (1 + \sqrt{5})/2$, we get

$$m \leq \log_\phi 2|\alpha_u - \beta_u|^{-1}; \tag{22}$$

comparing with (12) we observe that we might have more inversion transformations now. Summing this for all leaves gives us the following bound on the total number of inversion transformations in T'

$$\sum_{u\in U}(2 + \log_\phi |\alpha_u - \beta_u|^{-1}) \leq 2n + \sum_{u\in U} \log_\phi |\alpha_u - \beta_u|^{-1}. \tag{23}$$

5.2. Bounding the scalings and Taylor shifts

As was the case earlier, the purpose of scalings and Taylor shifts is to compute the floor of the smallest positive root of a polynomial. Using property (8) we will see that scaling the polynomial reduces the number of Taylor shifts required for our purpose. The key lemma is the following.

Lemma 5.2. *Let $B_1(X) \in \mathbb{R}[X]$. For $i > 1$ recursively define*

$$B_i(X) := \begin{cases} B_{i-1}(\mathrm{PLB}(B_{i-1})(X + 1)) & \text{if } \mathrm{PLB}(B_{i-1}) \geq 2 \\ B_{i-1}(X + 1) & \text{otherwise.} \end{cases}$$

Let $\alpha_i := \kappa(B_i(X))$. Then the number of scalings and Taylor shifts needed to yield $\alpha_i \leq 1$ are bounded by $O(\log n)$ and $O(n)$ respectively.

Proof. Let $b_i := \mathrm{PLB}(B_i)$. Suppose $\alpha_1 \geq 16n$ then from (8) we know that $b_1 \geq \frac{\alpha_1}{16n} \geq 1$. In this case, $\alpha_2 := \frac{\alpha_1}{b_1} - 2 < 16n$. Starting from α_2 we either perform a scaling by at least 2, and hence they are bounded by $\log 16n$, or a Taylor shift by at least one, which are hence bounded by $16n$. □

Remark 5.3. The purport of the above lemma and (22) is that we decrease the number of Taylor shifts done between consecutive inversion transformations, but this comes at the expense of increasing the number of inversion transformations. However, this is not a drawback, because each inversion transformation decreases the width of the associated interval by at least half.

Based upon this lemma we will bound the number of *scalings and Taylor shifts* to a leaf u of T'. Let $[q_0, \frac{p_1}{q_1}, \dots, \frac{p_m}{q_m}]$ be the continued fraction associated with u, and $M_u(X)$, I_u be the corresponding Möbius transformation and interval associated with u. Further, let $A_i(X)$ be the polynomial obtained after doing i inversion transformations along the path to u.

Again consider the situation shown in Fig. 4, i.e., while constructing q_i and p_{i+1} we have one type-2 node w_1 and one type-1 node u_1. From Lemma 5.2 we get that the number of scalings and Taylor shifts needed to reach w_1, from w_1 to u_1, and from u_1 is bounded by $O(\log n)$ and $O(n)$ respectively. In general, if we have k_i type-1 node, and ℓ_i type-2 nodes then the total number of scalings needed to construct $A_{i+1}(X)$ from $A_i(X)$ is bounded by $O(\log n(k_i + \ell_i + 1))$, and the total number of Taylor shifts needed for the same purpose is bounded by $O(n(k_i + \ell_i + 1))$. Summing this over $i = 1, \dots, m$ we get that the number of scalings along the path to u is bounded by $O(\sum_{i=1}^{m}(k_i + \ell_i) \log n + m \log n)$ and the number of Taylor shifts along the path to u is bounded by $O(\sum_{i=1}^{m} n(k_i + \ell_i) + nm)$. Summing these two bounds over all the leaves in T', along with (23) and the observation that the total number of type-1 and type-2 nodes in the tree are $O(n)$, we get the following result.

Lemma 5.4. *The number of scalings in the tree T' is bounded by*

$$O\left(n\log n + \log n \sum_{u\in U} \log|\alpha_u - \beta_u|^{-1}\right)$$

and the number of Taylor shifts is bounded by

$$O\left(n^2 + n\sum_{u\in U} \log|\alpha_u - \beta_u|^{-1}\right).$$

Clearly, this bound applies to $\#(T')$, and hence to $\#(T)$.

If we consider the graph whose edge set are the pairs (α_u, β_u), $u \in U$, and reorder the edges as was done in Section 3.3 then from Proposition 3.6 and Theorem 5.1 we get the following result.

Theorem 5.5. *Let $A_{in}(X) \in \mathbb{R}[X]$ be a square-free polynomial of degree n. Then in the recursion tree of the procedure CFS applied to $A_{in}(X)$, the number of nodes at which scalings are performed is*

$$O([n\log \mathrm{M}(A_{in}) + n\log n - \log|\mathrm{discr}(A_{in})| - n\log|\mathrm{lead}(A_{in})|]\log n),$$

and the number of nodes at which Taylor shifts are performed is

$$O(n^2\log \mathrm{M}(A_{in}) + n^2\log n - n\log|\mathrm{discr}(A_{in})| - n^2\log|\mathrm{lead}(A_{in})|).$$

For the special case of integer polynomials we get:

Corollary 5.6. *Let $A_{in}(X)$ be a square-free polynomial of degree n with integer coefficients of bit-length L. Then in the recursion tree of the procedure CFS applied to $A_{in}(X)$, the number of nodes at which scalings are performed is $\widetilde{O}(nL)$, and the number of nodes at which Taylor shifts are performed is $\widetilde{O}(n^2L)$.*

5.3. The bit-complexity

We proceed similar to Section 4, i.e., let $A_{\mathrm{in}}(X)$ be a square-free integer polynomial such that $\|A_{\mathrm{in}}\|_\infty < 2^L$. We will bound the worst case complexity of a node in the recursion tree of the procedure CFS applied to $A_{\mathrm{in}}(X)$.

Recall the definition of the tree T' and the set U of its leaves from the starting of this section. Let $u \in U$ be a leaf in T' with the associated continued fraction $[q_0, \frac{p_1}{q_1}, \ldots, \frac{p_{m+1}}{q_{m+1}}]$. Further, let $\|A_i\|_\infty < 2^{L_i}$, where $A_i(X)$ is the polynomial obtained after doing i inversion transformations along the path to u.

In order to construct $A_{i+1}(X)$ from $A_i(X)$ we need to construct a sequence of polynomials $B_j(X)$, $1 \le j \le \ell$, such that $B_1(X) := A_i(X)$ and $B_j(X)$ is constructed from $B_{j-1}(X)$ as in Lemma 5.2. Suppose in constructing this sequence, we do s_i scalings by amounts $\delta_1, \ldots, \delta_{s_i} \ge 2$, and t_i Taylor shifts by one. Since each scaling by δ_j is followed by a Taylor shift by one, the bit-length increases by $O(n(1 + \log\delta_j))$. Recall from Section 2.1 that the δ_j's are all powers of two and hence a scaling by δ_j is a simple bit-shift operation. Thus the bit-length of the polynomial $B_\ell(X)$ is bounded by $O(L_i + n\sum_{j=1}^{s_i}\log\delta_j + n(s_i + t_i))$, and hence the worst case complexity of computing $A_i(X)$ from $B_\ell(X)$ using classical Taylor shifts is bounded by $O(n^2(L_i + n\sum_{j=1}^{s_i}\log\delta_j + n(s_i + t_i)))$. Note that $\prod_{j=1}^{s_i}\delta_j = p_{i+1}$. Thus the worst case complexity of computing $A_{i+1}(X)$ from $A_i(X)$ is bounded by

$$O(n^2(L_i + n\log p_{i+1} + n(s_i + t_i))).$$

Summing this for $i = 1, \ldots, m$ we get that it is bounded by

$$O\left(n^2\left(L + n\log\prod_{i=1}^{m} p_{i+1} + n\sum_{i=1}^{m}(s_i + t_i)\right)\right). \tag{24}$$

The summation in the last term is a bound on the total number of Taylor shifts and scalings along the path in the tree to the leaf u, which we know from Corollary 5.6 is bounded by $\widetilde{O}(n^2L)$. Along with (20), we get that the worst case complexity of computing $A_{i+1}(X)$ is bounded by

$$\widetilde{O}(n^2(L + n\log Q_m + n^3L)).$$

But from (21) we know that $Q_m \le 2p_1\ldots p_m Q_{m-1}^{-1}|\alpha_u - \beta_u|^{-1}$, and from (20) that $Q_{m-1} \ge \prod_{i=1}^{m-1} p_{i+1}$. Thus $Q_m \le 2q_0|\alpha_u - \beta_u|^{-1}$. Note that q_0 is bounded by the length of the rightmost path of the tree, which we know is bounded by the absolute value of the largest root of $A_{\mathrm{in}}(X)$; from Cauchy's bound [5, p. 350] we know that the latter quantity is $O(L)$. Along with Remark 3.7, we get that the worst case complexity of any node in the recursion tree is bounded by $\widetilde{O}(n^5L)$. Now we apply Corollary 5.6, and obtain our second main result.

Theorem 5.7. *Let $A(X)$ be a square-free integer polynomial of degree n with integer coefficients of bit-length L. Then the bit-complexity of isolating all the real roots of $A(X)$ using the procedure CFS, where we use classical Taylor shifts, is $\widetilde{O}(n^7L^2)$.*

5.4. A modification of CFS

As compared to the result in Lemma 3.10, the extra factor n in Corollary 5.6 appears because of the bound on the number of Taylor shifts by one in Lemma 5.2. This instead occurs when the value returned by PLB(A) is at most one. One way to remedy this is to replace line five of the procedure CFS(A, M) with the following:

> 5. **If** $b := \mathrm{PLB}(A) > 1$ **then**
> $A(X) := A(b(X+1)), M(X) := M(b(X+1))$.
> **else**
> $A(X) := A(2X), M(X) := M(2X)$.

Let MCFS(A, M) be the modified procedure. We claim that the size of the recursion tree for this formulation is $\widetilde{O}(nL)$.

The continued fraction associated with a node in the recursion tree of the formulation is still of the form $[q_0, \frac{p_1}{q_1}, \ldots, \frac{p_m}{q_m}]$; however, instead of (18), now it satisfies

$$2q_i \geq p_{i+1}. \tag{25}$$

To see this, suppose that we have already done $(i-1)$ inversion transformations and are going to construct q_i and p_{i+1}; initially, i.e., just after the $(i-1)$th inversion transformation, both are one. Now, to construct them we perform scalings and Taylor shifts until we do an inversion transformation. Each scaling is followed by a Taylor shift by one, except possibly the scaling just before performing the ith inversion transformation; until that point the scenario is the same as for (18) and hence $q_i \geq p_{i+1}$; however, if the last scaling was by two and we did not do a Taylor shift by one subsequently then p_{i+1} has been doubled and q_i remains unchanged; but clearly, $2q_i \geq p_{i+1}$.

Let T be the recursion tree of the procedure MCFS initiated on the square-free polynomial $A_{\mathrm{in}}(X)$ and the Möbius transformation X. Construct the tree T', as was described in Section 3, from T. The termination criteria, Theorem 2.5 and Theorem 2.6, still hold and T' satisfies Theorem 5.1.

Now, we proceed to bound the number of inversion transformations in T'. The argument in Section 5.1 fails, because the inequality (20) no longer holds. Instead we have the following result.

Lemma 5.8. *Let* $[q_0, \frac{p_1}{q_1}, \frac{p_2}{q_2}, \ldots, \frac{p_m}{q_m}]$ *be a continued fraction satisfying* (25)*. Then*

$$Q_m Q_{m-1} \geq (3/2)^m \prod_{i=1}^{m} p_i.$$

Proof. Assume that $m = 2k$; a similar argument holds when m is odd. From (1) we know that

$$\frac{Q_{2k}}{p_{2k}} = Q_{2k-2} + \frac{q_{2k}}{p_{2k}}(p_{2k-1}Q_{2k-3} + q_{2k-1}Q_{2k-2}) \geq Q_{2k-2}\left(1 + \frac{q_{2k}}{p_{2k}}q_{2k-1}\right) \geq \frac{3}{2}Q_{2k-2},$$

because of (25) and since $q_{2k} \geq 1$. Thus recursively we get $Q_{2k} \geq (3/2)^k \prod_{i=1}^{k} p_{2i}$. We can similarly show that $Q_{2k-1} \geq (3/2)^k \prod_{i=1}^{k} p_{2i-1}$. Multiplying these two inequalities yields us the desired result. □

From this result it is clear that (23) holds with some constant multiplicative factor.

The number of scalings and Taylor shifts are now bounded by $O(\log n)$; this is clear from Lemma 5.2, and the observation that now each Taylor shift by one is preceded by a scaling by at least two. Based upon these results and following the arguments in Section 5.2, we get the following results.

Theorem 5.9. *The number of nodes in the recursion tree of the procedure MCFS applied to a square-free polynomial* $A_{in}(X) \in \mathbb{R}[X]$ *of degree* n *is bounded by*

$$O([n \log \mathrm{M}(A_{in}) + n \log n - \log|\mathrm{discr}(A_{in})| - n \log|\mathrm{lead}(A_{in})|] \log n).$$

For the special case of integer polynomials we get:

Corollary 5.10. *The number of nodes in the recursion tree of the procedure MCFS applied to a square-free polynomial of degree* n *with integer coefficients of bit-length* L *is bounded by* $\widetilde{O}(nL)$.

And finally from the arguments in Section 5.3 we have:

Theorem 5.11. *Let* $A(X)$ *be a square-free integer polynomial of degree* n *with integer coefficients of bit-length* L*. Then the bit-complexity of isolating all the real roots of* $A(X)$ *using the procedure MCFS, where we use classical Taylor shifts, is* $\widetilde{O}(n^5L^2)$.

6. Conclusion

We have derived bounds on the worst case complexity of three different formulations of the continued fraction algorithm. The key step in deriving these bounds is to bound the number of Taylor shifts done in the recursion tree of the algorithm. For the formulation by Akritas and the formulation by Akritas and Strzeboński, this bound is $\widetilde{O}(n^2L)$. The latter formulation, though, has a bias towards performing more inversion transformations, which substantiates its efficiency in practice; also, the introduction of scaling the polynomial, instead of directly doing a Taylor shift, reduces the worst case bit-complexity of this formulation by a factor of $\widetilde{O}(nL)$. Nevertheless, the additional factor of n in Corollary 5.6 prevents us from achieving the same complexity as that for Collins and Akritas' algorithm, which we know (e.g., see [15, Thm. 4.1]) is $\widetilde{O}(n^5L^2)$, assuming we use classical Taylor shifts. To close this gap, we proposed a further modification to the formulation of Akritas and Strzeboński. Though this formulation seems promising in theory, its performance in practice has not been tested (unlike the other two formulations). We surmise that the practical behaviour of this formulation is similar to Collins and Akritas' algorithm, because it has a greater bias towards performing more inversion transformations, which guarantee that the width of the associated interval goes down by at least half.

Another approach to close this complexity gap (instead of modifying the algorithm) is to use an almost ideal PLB function, i.e., that satisfies a tighter inequality than (4), which is satisfied by the bound by Hong. For instance, using Aberth's method [1,10] to approximate the roots of the polynomial to some absolute precision while only using floating-point arithmetic.

One may also try to show that the bounds in Corollaries 3.9 and 5.6 on the size of the recursion tree are essentially tight. This might be done by trying to construct a degree n polynomial with integer coefficients of bit-length L such that it has a pair of real roots α, β, whose separation is the worst case separation bound, and the continued fraction separating them is of the form $[q_0, \ldots, q_m]$ where $q_i = n$, for $i = 0, \ldots, m$, and $m = \Omega(nL/\log n)$; thus we can verify that $\prod_{i=1}^{m} q_i = O(|\alpha - \beta|^{-1})$. This would ensure that in the worst case the length of the path from the root of the recursion tree to the node separating α and β is $nm = \Omega(n^2L/\log n)$. Though we have failed to construct such an example, this attempt suggests that bounds in the two corollaries may be tight.

Acknowledgements

The author is indebted to Arno Eigenwillig, Victor Pan and to the anonymous referees for their suggestions.

References

[1] O. Aberth, Iteration methods for finding all zeros of a polynomial simultaneously, Mathematics and Computation 27 (1973) 339–344.
[2] A. Akritas, A new look at one of the Bisection Methods derived from Vincent's theorem or There is no Descartes' method. http://inf-server.inf.uth.gr/~akritas/articles/71.pdf.
[3] A. Akritas, A. Bocharov, A. Strzeboński, Implementation of real root isolation algorithms in Mathematica, in: Abstracts of the International Conference on Interval and Computer-Algebraic Methods in Science and Engineering, Interval'94, St. Petersburg, Russia, March 7–10 1994, pp. 23–27.
[4] A.G. Akritas, Vincent's theorem in algebraic manipulation, Ph.D. Thesis, Operations Research Program, North Carolina State University, Raleigh, NC, 1978.
[5] A.G. Akritas, Elements of Computer Algebra with Applications, John Wiley Interscience, New York, 1989.
[6] A.G. Akritas, A. Strzeboński, P. Vigklas, Improving the performance of the continued fractions method using new bounds of positive roots, Nonlinear Analysis: Modelling and Control 13 (3) (2008) 265–279.
[7] A.G. Akritas, A. Strzeboński, P. Vigklas, Implementations of a new theorem for computing bounds for positive roots of polynomials, Computing 78 (2006) 355–367. Published online.
[8] A. Alesina, M. Galuzzi, A new proof of Vincent's theorem, L'Enseignement Mathémathique 44 (1998) 219–256.
[9] S. Basu, R. Pollack, M.-F. Roy, Algorithms in Real Algebraic Geometry, 2nd edition, Springer, 2006.
[10] D.A. Bini, G. Fiorentino, Design, analysis, and implementation of a multiprecision polynomial rootfinder, Numerical Algorithms 23 (2000) 127–173. Package available at: http://www.dm.unipi.it/cluster-pages/mpsolve/index.htm.
[11] G.E. Collins, A.G. Akritas, Polynomial real root isolation using Descartes' rule of signs, in: R.D. Jenks (Ed.), Proc. of the 1976 ACM Symp. on Symbolic and Algebraic Computation, ACM Press, 1976, pp. 272–275.
[12] J.H. Davenport, Computer algebra for cylindrical algebraic decomposition, Technical Report, The Royal Institute of Technology, Department of Numerical Analysis and Computing Science, S-100 44, Stockholm, Sweden, 1985. Reprinted as: Tech. Rep. 88-10, School of Math. Sciences, U. of Bath, Bath, England. http://www.bath.ac.uk/~masjhd/TRITA.pdf.
[13] Z. Du, V. Sharma, C. Yap, Amortized bounds for root isolation via Sturm sequences, in: D. Wang and L. Zhi (Eds.), Proc. Int'l Workshop on Symbolic-Numeric Computation, School of Science, Beihang University, Beijing, China, 2005, pp. 81–93.
[14] A. Eigenwillig, Real root isolation for exact and approximate polynomials using Descartes' rule of signs, Ph.D. Thesis, Universität des Saarlandes, Saarbrücken, Germany, 2008.
[15] A. Eigenwillig, V. Sharma, C. Yap, Almost tight complexity bounds for the Descartes method, in: Proc. 2006 Int'l Symp. Symbolic and Algebraic Computation, ISSAC'06, Genova, Italy, 2006.
[16] I.Z. Emiris, B. Mourrain, E.P. Tsigaridas, Real algebraic numbers: Complexity analysis and experimentation, in: P. Hertling, C.M. Hoffmann, W. Luther, N. Revol (Eds.), Reliable Implementation of Real Number Algorithms: Theory and Practice, in: Lecture Notes in Computer Science, vol. 5045, Springer, 2006. Available as Research Report 5897, INRIA. http://www.inria.fr/rrrt/rr-5897.html.
[17] H. Hong, Bounds for absolute positiveness of multivariate polynomials, Journal of Symbolic Computation 25 (5) (1998) 571–585.
[18] A.Ya. Khinchin, Continued Fractions, Dover Publications, 1997.
[19] J. Kioustelidis, Bounds for the positive roots of the polynomials, Journal of Computational and Applied Mathematics 16 (1986) 241–244.
[20] W. Krandick, Isolierung reeller Nullstellen von Polynomen, in: J. Herzberger (Ed.), Wissenschaftliches Rechnen, Akademie-Verlag, Berlin, 1995, pp. 105–154.
[21] W. Krandick, K. Mehlhorn, New bounds for the Descartes method, Journal of Symbolic Computation 41 (1) (2006) 49–66.
[22] M. Mignotte, Some inequalities about univariate polynomials, in: Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation, SYMSAC 1981, ACM, 1981, pp. 195–199.
[23] M. Mignotte, D. Ştefănescu, Polynomials: An Algorithmic Approach, Springer, Singapore, 1999.

[24] N. Obreschkoff, Über die Wurzeln von algebraischen Gleichhungen, Jahresbericht der Deutschen Mathematiker-Vereinigung 33 (1925) 52–64.
[25] A. Ostrowski, Note on Vincent's theorem, The Annals of Mathematics 52 (3) (1950) 702–707.
[26] K. Roth, Rational approximations to algebraic numbers, Mathematika 2 (1955) 160–167.
[27] S.M. Rump, Polynomial minimum root separation, Math. Comp. 33 (1979) 327–336.
[28] V. Sharma, Complexity analysis of algorithms in algebraic computation, Ph.D. Thesis, Dept. of Computer Science, NYU, January 2007.
[29] V. Sharma, Complexity of real root isolation using continued fractions, in: C. Brown (Ed.), Proc. International Symposium on Symbolic and Algebraic Computation, ACM, 2007, pp. 339–346.
[30] D. Ştefănescu, New bounds for the positive roots of polynomials, Journal of Universal Computer Science 11 (12) (2005) 2132–2141.
[31] E.P. Tsigaridas, I.Z. Emiris, Univariate polynomial real root isolation: Continued fractions revisited, in: Y. Azar, T. Erlebach (Eds.), Proc. 13th European Symp. on Algorithms, ESA, in: LNCS, vol. 4168, Springer, 2006, pp. 817–828.
[32] E.P. Tsigaridas, I.Z. Emiris, On the complexity of real root isolation using continued fractions, Theoretical Computer Science 392 (1–3) (2008) 158–173.
[33] J.V. Uspensky, Theory of Equations, McGraw-Hill, New York, 1948.
[34] A. Vincent, Sur la résolution des équations numériques, Journal of Mathematics Pures Applications 1 (1836) 341–372.
[35] J. von zur Gathen, J. Gerhard, Fast algorithms for Taylor shifts and certain difference equations, in: Proc. 1997 Int'l Symp. on Symbolic and Algebraic Computation, ISSAC 1997, ACM, 1997, pp. 40–47.
[36] C.K. Yap, Fundamental Problems of Algorithmic Algebra, Oxford University Press, 2000.