Improved complexity bounds for real root isolation using Continued Fractions

Elias Tsigaridas

Improved bounds for Continued Fractions variants for real root isolation

Elias P. Tsigaridas
Computer Science Department, Aarhus University, Denmark
elias (AT) cs.au.dk

Abstract

We consider the problem of isolating the real roots of a square-free polynomial with integer coefficients using (variants of) the continued fraction algorithm (CF). We introduce a novel way to compute a lower bound on the positive real roots of univariate polynomials. This allows us to derive a worst case bound of $\widetilde{\mathcal{O}}_B(d^6 + d^4\tau^2 + d^3\tau^2)$ for isolating the real roots of a polynomial with integer coefficients using the classic variant [1] of CF, where d is the degree of the polynomial and τ the maximum bitsize of its coefficients. This improves the previous bound of Sharma [30] by a factor of d^3 and matches the bound derived by Mehlhorn and Ray [19] for another variant of CF; it also matches the worst case bound of the subdivision-based solvers.

1 Introduction

The problem of isolating the real roots of a square-free polynomial with integer coefficients is one of the most well-studied problems in symbolic computation and computational mathematics. The goal is to compute intervals with rational endpoints that contain one and only one real root of the polynomial, and to have one interval for every real root.

If we restrict ourselves to algorithms that perform computations with rational numbers of arbitrary size, then we can distinguish two main categories. The first one consists of algorithms that are subdivision-based; their process mimics binary search. They bisect an initial interval that contains all the real roots until they obtain intervals with one or zero real roots. The different variants differ in the way that they count the number of real roots inside an interval, for example using Sturm's theorem or Descartes' rule of signs, see also Th. 1. Classical representatives are the algorithms STURM, DESCARTES and BERNSTEIN. We refer the reader to [11, 12, 9, 17, 16, 15, 10, 26] and references therein for further details. The worst case complexity of all variants in this category is $\widetilde{\mathcal{O}}_B(d^6 + d^4\tau^2)$, where d is the degree of the polynomial and τ the maximum bitsize of its coefficients. Especially, for the STURM solver, recently, it was proved that its expected case complexity, if we consider certain random polynomials as input, is $\widetilde{\mathcal{O}}_B(r\, d^2\tau)$, where r is the number of real roots [13]. Let us also mention the bitstream version of DESCARTES algorithm, cf. [20] and references therein.

The second category contains algorithms that isolate the real roots of a polynomial by computing their continued fraction expansion (CF). Since successive approximants of a real number define an interval that contains this number, CF computes the partial quotients of the roots of the polynomial until the corresponding approximants correspond to intervals that isolate the real roots. Counting of the real roots is based on Descartes' rule of signs (Th. 1) and termination is guaranteed by Vincent's theorem (Th. 3). There are several variants which they differ in the way that they compute the partial quotients.

The first formulation of the algorithm is due to Vincent [35], who computed the partial quotients by successive transformations of the form $x \mapsto x + 1$. An upper bound on the number of partial quotients needed was derived by Uspensky [33]. Unfortunately this approach leads to an exponential complexity bound. Akritas [1], see also [3, 2], treated the exponential behavior of CF by treating the partial quotients as lower bounds of the positive real roots, and computed the bounds using Cauchy's bound. With this approach, c repeated operations of the form $x \mapsto x + 1$ could be replaced by $x \mapsto x + c$. However, his analysis assumes an ideal positive lower bound, that is that we can compute directly the floor of the smallest positive real root. In [31], it was proven, under the assumption that Gauss-Kuzmin distribution holds for the real algebraic numbers, that the expected complexity of CF is $\widetilde{\mathcal{O}}_B(d^4\tau^2)$. By spreading the roots, the expected complexity becomes $\widetilde{\mathcal{O}}_B(d^4 + d^3\tau)$ [32]. The first worst-case complexity result of CF, $\widetilde{\mathcal{O}}_B(d^8\tau^3)$, is due to Sharma [30], without any assumption. He also proposed a variant of CF, that combines continued fractions with subdivision, with complexity $\widetilde{\mathcal{O}}_B(d^5\tau^2)$. All the variants of CF in [30] compute lower bounds on the positive roots using Hong's bound [14], which is assumed to have quadratic arithmetic complexity. Mehlhorn and Ray [19] proposed a novel way of computing Hong's bound based on incremental convex hull computations with linear arithmetic complexity. A direct consequence is that they reduced the complexity of the variant of CF combined with subdivision [30] to $\widetilde{\mathcal{O}}(d^4\tau^2)$, thus matching the worst case complexity of the subdivision-based algorithms. Using [19] and fast Taylor shifts [36], the bound [30] on classical variant of CF becomes $\widetilde{\mathcal{O}}_B(d^7\tau^3)$.

As far as the numerical algorithms are concerned, the best known bound for the problem is due to Pan [23, 22] and Schönhage [28], see also [29], $\widetilde{\mathcal{O}}_B(d^3\tau)$. Moreover, it seems that Pan's approach could be improved to $\widetilde{\mathcal{O}}_B(d^2\tau)$. This class of algorithms approximate the roots, real and complex, of the input polynomials up to a precision. They could be turned to root isolation algorithms by requiring them to approximate up to the separation bound, that is the minimum distance between the roots. The crux of the algorithms is that they recursively split the polynomial until we obtain linear factors that approximate sufficiently all the roots, real and complex. We also refer to a recent approach that concentrates only on the real roots [24]. For an implementation of Schönhage's algorithm we refer the reader to the routine CPRTS, p.12 in Addenda, based on the multitape Turing machine[1]. We are not aware of any implementation of Pan's algorithm. In the special case where all the roots of the polynomial are real, also called the *real root problem*, dedicated numerical algorithms were proposed by Reif [25] and Ben-Or and Tiwari [6] for approximating the roots. Nevertheless, their Boolean complexity is also $\widetilde{\mathcal{O}}_B(d^3\tau)$. Quite recently, Sagraloff [27] announced a variant of the bitstream version of DESCARTES algorithm with complexity $\widetilde{\mathcal{O}}_B(d^3\tau^2)$.

Our contribution. We present a novel way to compute a lower bound on the positive real roots of a univariate polynomial (Lem. 5). The proposed approach computes the floor of the root (possible complex) with the smallest positive real part that contributes to the number of the sign variations in the coefficients list of the polynomial. Our bound is at least as good as Hong's bound [14]. Using this lower bound we improve the worst case bit complexity bound of the classical variant of CF, obtained by Sharma [30], by a factor of d^3. We obtain a bound of $\widetilde{\mathcal{O}}_B(d^6 + d^4\tau^2)$ or $\widetilde{\mathcal{O}}_B(N^6)$, where $N = \max\{d, \tau\}$, (Th. 7), which matches the worst case bound of the subdivision-based solvers and also matches the bound due to Mehlhorn and Ray [19] achieved for another variant of CF; it also matches the worst case bound of the subdivision-based solvers [11, 12, 9, 17, 16, 15, 10, 26].

Notation. In what follows $\mathcal{O}_B$, resp. $\mathcal{O}$, means bit, resp. arithmetic, complexity and the $\widetilde{\mathcal{O}}_B$, resp. $\widetilde{\mathcal{O}}$, notation means that we are ignoring logarithmic factors. For a polynomial $A \in \mathbb{Z}[x]$, $\deg(A) = d$ denotes its degree and $\mathcal{L}(A) = \tau$ the maximum bitsize of its coefficients, including a bit for the sign. For $a \in \mathbb{Q}$, $\mathcal{L}(a) \geq 1$ is the maximum bitsize of the numerator and the denominator. Let $\mathsf{M}(\tau)$ denote the bit complexity of multiplying two integers of size τ; using FFT, $\mathsf{M}(\tau) = \widetilde{\mathcal{O}}_B(\tau)$. To simplify notation, we will assume throughout the paper that $\lg(\deg(A)) = \lg d = \mathcal{O}(\tau) = \mathcal{O}(\mathcal{L}(A))$. By $\texttt{VAR}(A)$ we denote the number of sign variations in the list of coefficients of A. We use Δ_γ to denote the minimum distance between a root γ of a polynomial A and any other root, we call this quantity *local separation bound*; $\Delta = \min_\gamma \Delta_\gamma$ is the *separation bound*, that is the minimum distance between all the roots of A.

[1] `http://www.iai.uni-bonn.de/~schoe/tp/TPpage.html`

2 A short introduction to continued fractions

Our presentation follows closely [32]. For additional details we refer the reader to, e.g., [37, 7, 34]. In general, a *simple (regular) continued fraction* is a (possibly infinite) expression of the form

$$q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \dots}} = [q_0, q_1, q_2, \dots],$$

where the numbers q_i are called *partial quotients*, $q_i \in \mathbb{Z}$ and $q_i \geq 1$ for $i > 0$. Notice that q_0 may have any sign, however, in our real root isolation algorithm $q_0 \geq 0$, without loss of generality. By considering the recurrent relations

$$\begin{array}{lll} P_{-1} = 1, & P_0 = q_0, & P_{n+1} = q_{n+1}\, P_n + P_{n-1}, \\ Q_{-1} = 0, & Q_0 = 1, & Q_{n+1} = q_{n+1}\, Q_n + Q_{n-1}, \end{array} \tag{1}$$

it can be shown by induction that $R_n = \frac{P_n}{Q_n} = [q_0, q_1, \dots, q_n]$, for $n = 0, 1, 2, \dots$.

If $\gamma = [q_0, q_1, \dots]$ then $\gamma = q_0 + \frac{1}{Q_0 Q_1} - \frac{1}{Q_1 Q_2} + \dots = q_0 + \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{Q_{n-1} Q_n}$ and since this is a series of decreasing alternating terms it converges to some real number γ. A finite section $R_n = \frac{P_n}{Q_n} = [q_0, q_1, \dots, q_n]$ is called the $n-$th *convergent* (or *approximant*) of γ and the tails $\gamma_{n+1} = [q_{n+1}, q_{n+2}, \dots]$ are known as its *complete quotients*. That is $\gamma = [q_0, q_1, \dots, q_n, \gamma_{n+1}]$ for $n = 0, 1, 2, \dots$. There is an one to one correspondence between the real numbers and the continued fractions, where evidently the finite continued fractions correspond to rational numbers.

It is known that $Q_n \geq F_{n+1}$ and that $F_{n+1} < \phi^n < F_{n+2}$, where F_n is the $n-$th Fibonacci number and $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio. Continued fractions are the best rational approximation (for a given denominator size). This is as follows: $\frac{1}{Q_n(Q_{n+1}+Q_n)} \leq \left| \gamma - \frac{P_n}{Q_n} \right| \leq \frac{1}{Q_n Q_{n+1}} < \phi^{-2n+1}$.

In order to indicate or to emphasize that a partial quotient or an approximant belong to a specific real number γ, we use the notation q_i^{γ} and $R_n^{\gamma} = P_n^{\gamma}/Q_n^{\gamma}$, respectively. We also use $q_i^{(k)}$ and $R_n^{(k)} = P_n^{(k)}/Q_n^{(k)}$, where k is a non-negative integer, to indicate that we refer to the (real part of the) root γ_k of a polynomial A. The ordering of the roots is considered with respect to the magnitude of their (positive) real part.

3 Worst case complexity of CF

Theorem 1 (Descartes' rule of sign). *The number R of real roots of $A(x)$ in $(0, \infty)$ is bounded by $\mathtt{VAR}(A)$ and we have $R \equiv \mathtt{VAR}(A) \mod 2$.*

Remark 2. *In general Descartes' rule of sign obtains an overestimation of the number of the positive real roots. However, if we know that A is* hyperbolic, *i.e. has only real roots, or when the number of sign variations is 0 or 1 then it counts exactly.*

The CF algorithm depends on the following theorem, which dates back to Vincent's theorem in 1836 [35]. The inverse of Th. 3 can be found in [4, 8, 21]. The version of the theorem that we present is due to Alesina and Galuzzi [5], see also [33, 1, 4, 2], and its proof is closely connected to the one and two circle theorems (refer to [17, 5] and references therein).

Theorem 3. [5] *Let $A \in \mathbb{Z}[x]$ be square-free and let $\Delta > 0$ be the separation bound, i.e. the smallest distance between two (complex) roots of A. Let n be the smallest index such that $F_{n-1}\, F_n\, \Delta > \frac{2}{\sqrt{3}}$, where F_n is the n-th Fibonacci number. Then the map $x \mapsto [c_0, c_1, \ldots, c_n, x]$, where $c_0, c_1, \ldots, c_n$ is an arbitrary sequence of positive integers, transforms $A(x)$ to $A_n(x)$, whose list of coefficients has no more than one sign variation.*

For a polynomial $A = \sum_{i=0}^{d} a_i x^i$, where γ correspond to its (complex) roots, the Mahler measure, $\mathcal{M}(A)$, of A is $\mathcal{M}(A) = a_d \prod_{|\gamma| \geq 1} |\gamma|$, e.g. [21, 37]. If we further assume that $A \in \mathbb{Z}[x]$ and $\mathcal{L}(A) = \tau$ then $\mathcal{M}(A) \leq \|A\|_2 \leq \sqrt{d+1}\|A\|_\infty = 2^\tau \sqrt{d+1}$, and so $\prod_{|\gamma| \geq 1} |\gamma| \leq 2^\tau \sqrt{d+1}$.

We will also use the following aggregate bound. For a proof we refer to e.g. [32, 9, 10, 21, 15].

Theorem 4. *Let $A \in \mathbb{Z}[x]$ such that $\deg(A) = d$ and $\mathcal{L}(A) = \tau$. Let γ denotes its distinct roots, then*

$$\prod_{\gamma} \Delta_\gamma \geq 2^{-d^2}\, \mathcal{M}(A)^{-2d} \Leftrightarrow -\lg \prod_{\gamma} \Delta_\gamma = -\sum_{\gamma} \lg \Delta_\gamma \leq 3d^2 + 3d \lg d + 3d\tau.$$

3.1 The tree

The CF algorithm relies on Vincent's theorem (Th. 3) and Descartes' rule of sign (Th. 1) to isolate the positive real roots of a square-free polynomial A. The negative roots are isolated after we perform the transformation $x \mapsto -x$; hence it suffices to consider only the case of positive real roots throughout the analysis.

The pseudo-code of the classic variant of CF is presented in Alg. 1.

Given a polynomial A, we compute the floor of the smallest positive real root (PLB = Positive Lower Bound). The *ideal* PLB 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.

Then we perform the transformation $x \mapsto x + \mathrm{b}$, obtaining a polynomial A_{b}. It holds that $\mathtt{VAR}(A) \geq \mathtt{VAR}(A_{\mathrm{b}})$. The latter polynomial is transformed to A_1 by the transformation $x \mapsto 1 + x$ and if $\texttt{VAR}(A_1) = 0$ or $\texttt{VAR}(A_1) = 1$, then A_{b} has 0, resp. 1, real roots greater than 1, or equivalently A has 0, resp. 1, real roots greater than $\mathsf{b} + 1$ (Th. 1). If $\texttt{VAR}(A_1) < \texttt{VAR}(A_{\mathsf{b}})$ then (possibly) there are real roots of A_{b} in $(0, 1)$, or equivalently, there are real roots of A in $(\mathsf{b}, \mathsf{b} + 1)$, due to Budan's theorem. We apply the transformation $x \mapsto 1/(1 + x)$ to A_{b}, and we get the polynomial A_2. If $\texttt{VAR}(A_2) = 0$ or $\texttt{VAR}(A_2) = 1$, A_{b} has 0, resp. 1, real root less than 1 (Th. 1), or equivalently A has 0, resp. 1, real root less than $\mathsf{b} + 1$, or to be more specific in $(\mathsf{b}, \mathsf{b} + 1)$ (Th. 1). If the transformed polynomial, A_1 and A_2, have more than one sign variations, then we apply PLB to them and we repeat the process.

Following [1, 32, 30] we consider the process of the algorithm as an infinite binary tree. The nodes of the tree hold polynomials and (isolating) intervals. The root of the tree corresponds to the original polynomial A and the shifted polynomial A_{b}. The branch from a node to a right child corresponds to the map $x \mapsto x + 1$, which yields polynomial A_1, while to the left child to the map $x \mapsto 1/(1 + x)$, which yields polynomial A_2. The sequence of transformations that we perform is equivalent to the sequence of transformations in Th. 3, and so the leaves of the tree hold (transformed) polynomials that have no more than one sign variation, if Th. 3 holds.

A polynomial that corresponds to a leaf of the tree and has one sign variation it is produced after a transformation as in Th. 3, using positive integers $q_0, q_1, \ldots, q_n$. The compact form of this is $M : x \mapsto \frac{P_n x + P_{n-1}}{Q_n x + Q_{n-1}}$, where $\frac{P_{n-1}}{Q_{n-1}}$ and $\frac{P_n}{Q_n}$ are consecutive convergents of the continued fraction $[q_0, q_1, \ldots, q_n]$. The polynomial has one real root in $(0, \infty)$, thus the (unordered) endpoints of the isolating interval are $M(0) = \frac{P_{n-1}}{Q_{n-1}}$ and $M(\infty) = \frac{P_n}{Q_n}$.

There are different variants of the algorithm that differ in the way they compute PLB. A PLB realization that actually computes exactly the floor of the smallest positive real root is called *ideal*, but unfortunately has a prohibitive complexity.

A crucial observation is that Descartes' rule of sign (Th. 1), that counts the number of sign variations depends not only on positive real roots, but also on some complex ones; which have positive real part. Roughly speaking CF is trying to isolate the positive real parts of the roots of A that contribute to the sign variations. Thus, the ideal PLB suffices to compute the floor of the smallest positive real part of the roots of A that contribute to the number of sign variations. For this we will use Lem. 5. Notice that all the positive real roots contribute to the number of sign variation of A, but this is not always the case for the complex roots with positive real part.

3.2 Computing a partial quotient

Lemma 5. *Let $A \in \mathbb{Z}[x]$, such that $\deg(A) = d$ and $\mathcal{L}(A) = \tau$. We can compute the first partial quotient, or in the other words the floor*[2]*, c, of the real part of the root with the smallest real part, that contributes to the sign variations of A in $\widetilde{\mathcal{O}}_B(d\tau \lg c + d^2 \lg^2 c)$.*

Proof: We compute the corresponding integer using the technique of the exponential search, see for example [18]. Without loss of generality, we may assume that the real root is not in $(0, 1)$, since in this case we should return 0.

We perform the transformation $x \mapsto x + 2^0$ to the polynomial, and then the transformation $x \mapsto x + 1$. If the number of sign variations of the resulting polynomial compared to the original one decreases, then $2^0 = 1$ is the partial quotient. If not, then we perform the transformation $x \mapsto x + 2^1$. If the number of sign variations does not decrease, then we perform $x \mapsto x + 2^2$. Again if the number of sign variations does not decrease, then we perform $x \mapsto x + 2^3$ and so on. Eventually, for some positive integer k, there would be a loss in the sign variations between transformations $x \mapsto x + 2^{k-1}$ and $x \mapsto x + 2^k$. In this case the partial quotient c, which we want to compute, satisfies $2^{k-1} < c < 2^k < 2\,c$. The exact value of c is computed by performing binary search in the interval $[2^k, 2^{k+1}]$. We deduce that the number of transformations that we need to perform is $2k + \mathcal{O}(1) = 2 \lg\lfloor c \rfloor + \mathcal{O}(1)$.

In the worst case, each transformation corresponds to an asymptotically fast Taylor shift with a number of bitsize $\mathcal{O}(\lg c)$, which costs[3] $\mathcal{O}_B(\mathsf{M}\left(d\tau + d^2 \lg c\right) \lg d)$ [36, Th. 2.4]. By considering fast multiplication algorithms the costs becomes $\widetilde{\mathcal{O}}_B(d\tau + d^2 \lg c)$ and multiplying by the number of transformations needed, $\lg c$, we conclude the proof.

It is worth noticing that we do not consider the cases $c = 2^k$ or $c = 2^{k+1}$, since then we have computed, exactly, a rational root. □

3.3 Shifts operations and total complexity

Up to some constant factors, we can replace Δ in Th. 3 by Δ_γ, refer to [30] for a proof. This allows us to estimate the number, m_γ, of partial quotients needed, in the worst case, to isolate the positive real part of a root γ. It holds

$$m_\gamma \leq \frac{1}{2}(1 + \log_\phi 2 - \lg \Delta_\gamma) \leq 2 - \frac{1}{2} \lg \Delta_\gamma.$$

[2] We choose to use c instead of q_0 because in the complexity analysis that follow A could be a result of a shift operation, thus the computed integer may not be the 0-th partial quotient of the root that we are trying to approximate.

[3] Following Th. 2.4(E) in [36] the cost of performing the operation $f(x + a)$, where $\deg(f) = n$, $\mathcal{L}(f) = \tau$ and $\mathcal{L}(a) = \sigma$ is $\mathcal{O}_B(\mathsf{M}\left(n\tau + n^2\sigma\right) \lg n))$, and if we assume fast multiplication algorithms between integers, then it becomes $\widetilde{\mathcal{O}}_B(n\tau + n^2\sigma)$.

The transformed polynomial has either one or zero sign variation and if $\gamma \in \mathbb{R}$, then the corresponding interval isolates γ from the other roots of A. The associated continued fraction of (the real part of) γ is $[q_0^\gamma, q_1^\gamma, \ldots, q_{m_\gamma}^\gamma]$. It holds that $\sum_\gamma m_\gamma = \mathcal{O}(d^2 + d\tau)$ [32, 30]. The following lemma bounds the bitsize of the partial quotients, q_k^γ, of a root γ.

Lemma 6. *Let $A \in \mathbb{Z}[x]$, such that $\deg(A) = d$ and $\mathcal{L}(A) = \tau$. For the real part of any root γ it holds*

$$\sum_{j=0}^{m_\gamma} \lg(q_j^\gamma) = \lg(q_0^\gamma) + \sum_{j=1}^{m_\gamma} \lg(q_j^\gamma) \le \lg(q_0^\gamma) + 1 - \lg \Delta_\gamma,$$

where we assume that $q_0^\gamma > 0$, and the term $1 - \lg \Delta_\gamma$ appears only when $\Delta_\gamma < 1$, i.e. when $m_\gamma \ge 1$. Moreover $\sum_\gamma \lg(q_0^\gamma) \le \lg \|A\|_2 \le \tau + \lg d$ and if γ ranges over a subset of distinct roots of A, then

$$\sum_\gamma \sum_{k=0}^{m_\gamma} \lg q_k^\gamma \le 1 + \tau + \lg d - \lg \prod_\gamma \Delta_\gamma = \mathcal{O}(d^2 + d\tau).$$

Proof: The Mahler measure, $\mathcal{M}(A)$, of A is $\mathcal{M}(A) = a_d \prod_{|\gamma| \ge 1} |\gamma|$. It also holds $\mathcal{M}(A) \le \|A\|_2 \le \sqrt{d+1}\|A\|_\infty = 2^\tau \sqrt{d+1}$, and so $\prod_{|\gamma| \ge 1} |\gamma| \le 2^\tau \sqrt{d+1}$. Since q_0^γ is the integer part of γ it holds $\prod_\gamma q_0^\gamma \le \prod_{|\gamma| \ge 1} |\gamma| \le \|A\|_2$ and thus

$$\sum_\gamma \lg(q_0^\gamma) \le \lg \sqrt{d+1} + \lg \|A\|_\infty \le \tau + \lg d. \tag{2}$$

Following [30] we know that

$$\frac{1}{Q_{m_\gamma}^\gamma Q_{m_\gamma - 1}^\gamma} \ge \frac{\Delta_\gamma}{2} \Leftrightarrow Q_{m_\gamma}^\gamma Q_{m_\gamma - 1}^\gamma \le 2/\Delta_\gamma. \tag{3}$$

From Eq. (1) we get $Q_k = q_k Q_{k-1} + Q_{k-2} \Rightarrow Q_k \ge q_k Q_{k-1}$, for $k \ge 1$. I we apply the previous relation recursively we get $\prod_{k=1}^{m_\gamma} q_k^\gamma \le Q_{m_\gamma}^\gamma \le 2/\Delta_\gamma$ and $\prod_{k=1}^{m-1} q_k^\gamma \le Q_{m_\gamma - 1}^\gamma \le 2/\Delta_\gamma$, and so

$$\sum_{k=1}^{m_\gamma} \lg q_k^\gamma = \lg \prod_{k=1}^{m} q_k^\gamma \le 1 - \lg \Delta_\gamma.$$

Finally, we sum over all roots γ and we use (2) and Th. 4,

$$\begin{aligned}\sum_\gamma \sum_{k=0}^{m_\gamma} \lg q_k^\gamma = \sum_\gamma \lg q_0^\gamma + \sum_\gamma \sum_{k=1}^{m_\gamma} \lg q_k^\gamma \le \sum_\gamma \lg q_0^\gamma + \sum_\gamma (1 - \lg \Delta_\gamma) \\ \le 1 + \tau + \lg d + d^2 + 3d \lg d + 3d\tau,\end{aligned}$$

which completes the proof. □

At each step of CF we compute a partial quotient and we apply a Taylor shift to the polynomial with this number. In the worst case we increase the bitsize of the polynomial by an additive factor of $\mathcal{O}(d \lg(q_k^\gamma))$, at each step. The overall complexity of CF is dominated by the computation of the partial quotients.

The following table summarizes the costs of computing the partial quotients of γ that we need:

$$
\begin{array}{ll}
0^{th} \text{ step} & \widetilde{\mathcal{O}}_B(d\tau \lg(q_0^\gamma) + d^2 \lg(q_0^\gamma) \lg(q_0^\gamma)) \\
1^{st} \text{ step} & \widetilde{\mathcal{O}}_B(d\tau \lg(q_1^\gamma) + d^2 \lg(q_0^\gamma q_1^\gamma) \lg(q_1^\gamma)) \\
 & \left(= \widetilde{\mathcal{O}}_B(d(\tau + d\lg(q_0^\gamma)) \lg(q_1^\gamma) + d^2 \lg^2(q_1^\gamma))\right) \\
2^{nd} \text{ step} & \widetilde{\mathcal{O}}_B(d\tau \lg(q_2^\gamma) + d^2 \lg(q_0^\gamma q_1^\gamma q_2^\gamma) \lg(q_2^\gamma)) \\
\vdots & \\
m_\gamma^{th} \text{ step} & \widetilde{\mathcal{O}}_B(d\tau \lg(q_m^\gamma) + d^2 \lg(\prod_{k=0}^{m} q_k^\gamma) \lg(q_m^\gamma))
\end{array}
$$

We sum over all steps to derive the cost for isolating γ, $\mathcal{C}^\gamma$, and after applying some obvious simplifications and use Lem. 6 we get

$$
\begin{aligned}
\mathcal{C}^\gamma &= \widetilde{\mathcal{O}}_B\left(d\tau \sum_{k=0}^{m_\gamma} \lg(q_k^\gamma) + d^2 \sum_{k=0}^{m_\gamma} \lg(q_k^\gamma) \lg \prod_{j=0}^{m_\gamma} q_j^\gamma \right) = \widetilde{\mathcal{O}}_B\left(d\tau \sum_{k=0}^{m_\gamma} \lg(q_k^\gamma) + d^2 \left(\sum_{k=0}^{m_\gamma} \lg(q_k^\gamma) \right)^2 \right) \\
&= \widetilde{\mathcal{O}}_B\left(d\tau (\lg(q_0^\gamma) - \lg \Delta_\gamma) + d^2 (\lg^2(q_0^\gamma) + \lg^2 \Delta_\gamma) \right).
\end{aligned}
$$

To derive the overall complexity, $\mathcal{C}$, we sum over all the roots that CF tries to isolate and we use Lem. 6 and Th. 4. Then

$$
\begin{aligned}
\mathcal{C} &= \textstyle\sum_\gamma \mathcal{C}^\gamma \\
&= \widetilde{\mathcal{O}}_B\left(d\tau \textstyle\sum_\gamma \lg(q_0^\gamma) - d\tau \sum_\gamma \lg \Delta_\gamma + d^2 \sum_\gamma \lg^2(q_0^\gamma) + d^2 \sum_\gamma \lg^2 \Delta_\gamma \right) \\
&= \widetilde{\mathcal{O}}_B\left(d\tau \textstyle\sum_\gamma \lg(q_0^\gamma) - d\tau \sum_\gamma \lg \Delta_\gamma + d^2 (\sum_\gamma \lg(q_0^\gamma))^2 + d^2 (\sum_\gamma \lg \Delta_\gamma)^2 \right) \\
&= \widetilde{\mathcal{O}}_B\left(d\tau(\tau + \lg d) + d\tau(d^2 + d \lg d + d\tau) + d^2(\tau^2 + \lg^2 d) + d^2(d^4 + d^2\tau^2) \right) \\
&= \widetilde{\mathcal{O}}_B(d^6 + d^4\tau^2).
\end{aligned} \tag{4}
$$

In the previous equation it possible to write $\sum_\gamma \lg^2 \Delta_\gamma \leq \left(\sum_\gamma \lg \Delta_\gamma\right)^2$ because $\Delta_\gamma < 1$, and hence $\lg \Delta_\gamma < 0$, for all γ that are involved in the sum. For the roots that holds $\Delta_\gamma \geq 1$ the algorithm isolates them without computing any of their partial quotients, with the exception of q_0^γ.

The previous discussion leads to the following theorem.

Theorem 7. *Let $A \in \mathbb{Z}[x]$, where $\deg(A) = d$ and $\mathcal{L}(A) = \tau$. The worst case complexity of isolating the real roots of A using the CF is $\widetilde{\mathcal{O}}_B(d^6 + d^4\tau^2)$.*

Algorithm 1: CF(A, M)

Input: $A \in \mathbb{Z}[X], M(X) = \frac{kX+l}{mX+n}$, $k, l, m, n \in \mathbb{Z}$
Output: A list of isolating intervals
Data: Initially $M(X) = X$, i.e. $k = n = 1$ and $l = m = 0$

if $A(0) = 0$ **then**
  OUTPUT Interval($M(0), M(0)$) ;
  $A \leftarrow A(X)/X$;
  CF(A, M);
$V \leftarrow$ Var(A);
if $V = 0$ **then** RETURN;
if $V = 1$ **then**
  OUTPUT Interval($M(0), M(\infty)$);
  RETURN;
$b \leftarrow$ PLB(A) // PLB $\equiv$ PositiveLowerBound ;
if $b \geq 1$ **then** $A_b \leftarrow A(b + X), M \leftarrow M(b + X)$;
$A_1 \leftarrow A_b(1 + X)$, $M_1 \leftarrow M(1 + X)$;
CF(A_1, M_1) // Looking for real roots in $(1, +\infty)$;
$A_2 \leftarrow A_b(\frac{1}{1+X})$, $M_2 \leftarrow M(\frac{1}{1+X})$;
CF(A_2, M_2) // Looking for real roots in $(0, 1)$;
RETURN;

References

[1] A. Akritas. An implementation of Vincent's theorem. *Numerische Mathematik*, 36:53–62, 1980.

[2] A. Akritas. There is no "Uspensky's method". Extended Abstract. In *Proc. Symp. on Symbolic and Algebraic Computation*, pages 88–90, Waterloo, Canada, 1986.

[3] A. Akritas, W. Strzeboński, and P. S. Vigklas. Implementations of a New Theorem for Computing Bounds for Positive Roots of Polynomials. *Computing*, 78:355–367, 2006.

[4] A.G. Akritas. *Elements of Computer Algebra with Applications*. J. Wiley & Sons, New York, 1989.

[5] A. Alesina and M. Galuzzi. A new proof of Vincent's theorem. *L' Enseignement Mathématique*, 44:219–256, 1998.

[6] M. Ben-Or and P. Tiwari. Simple algorithms for approximating all roots of a polynomial with real roots. *Journal of Complexity*, 6(4):417–442, 1990.

[7] E. Bombieri and A. van der Poorten. Continued fractions of algebraic numbers. In *Computational algebra and number theory (Sydney, 1992)*, pages 137–152. Kluwer Acad. Publ., Dordrecht, 1995.

[8] G.E. Collins and R. Loos. Real zeros of polynomials. In B. Buchberger, G.E. Collins, and R. Loos, editors, *Computer Algebra: Symbolic and Algebraic Computation*, pages 83–94. Springer-Verlag, Wien, 2nd edition, 1982.

[9] J. H. Davenport. Cylindrical algebraic decomposition. Technical Report 88–10, School of Mathematical Sciences, University of Bath, England, available at: http://www.bath.ac.uk/masjhd/, 1988.

[10] Z. Du, V. Sharma, and C. K. Yap. Amortized bound for root isolation via Sturm sequences. In D. Wang and L. Zhi, editors, *Int. Workshop on Symbolic Numeric Computing*, pages 113–129, School of Science, Beihang University, Beijing, China, 2005. Birkhauser.

[11] A. Eigenwillig, V. Sharma, and C. K. Yap. Almost tight recursion tree bounds for the Descartes method. In *Proc. ACM ISSAC*, pages 71–78, New York, USA, 2006.

[12] I. Z. Emiris, B. Mourrain, and E. P. Tsigaridas. Real Algebraic Numbers: Complexity Analysis and Experimentation. In P. Hertling, C. Hoffmann, W. Luther, and N. Revol, editors, *Reliable Implementations of Real Number Algorithms: Theory and Practice*, volume 5045 of *LNCS*, pages 57–82. Springer Verlag, 2008.

[13] Ioannis Z. Emiris, André Galligo, and Elias P. Tsigaridas. Random polynomials and expected complexity of bisection methods for real solving. In S. Watt, editor, *Proc. 35th ACM Int'l Symp. on Symbolic & Algebraic Comp. (ISSAC)*, pages 235–242, Munich, Germany, July 2010. ACM.

[14] H. Hong. Bounds for absolute positiveness of multivariate polynomials. *J. of Symbolic Computation*, 25(5):571–585, May 1998.

[15] J. R. Johnson. *Algorithms for Polynomial Real Root Isolation*. PhD thesis, The Ohio State University, 1991.

[16] W. Krandick. Isolierung reeller Nullstellen von Polynomen. In J. Herzberger, editor, *Wissenschaftliches Rechnen*, pages 105–154. Akademie-Verlag, Berlin, 1995.

[17] W. Krandick and K. Mehlhorn. New bounds for the Descartes method. *JSC*, 41(1):49–66, Jan 2006.

[18] S. Kwek and K. Mehlhorn. Optimal search for rationals. *Information Processing Letters*, 86(1):23–26, 2003.

[19] K. Mehlhorn and S. Ray. Faster algorithms for computing Hong's bound on absolute positiveness. *J. Symbolic Computation*, 45(6):677 – 683, 2010.

[20] Kurt Mehlhorn and Michael Sagraloff. A deterministic algorithm for isolating real roots of a real polynomial. *J. Symbolic Computation*, 46(1):70–90, 2011.

[21] M. Mignotte. *Mathematics for Computer Algebra.* Springer-Verlag, NY, 1991.

[22] V.Y. Pan. Solving a polynomial equation: Some history and recent progress. *SIAM Rev.*, 39(2):187–220, 1997.

[23] V.Y. Pan. Univariate polynomials: Nearly optimal algorithms for numerical factorization and rootfinding. *J. Symbolic Computation*, 33(5):701–733, 2002.

[24] V.Y. Pan, B. Murphy, R.E. Rosholt, G. Qian, and Y. Tang. Real rootfinding. *Proc. Int'l Work. on Symbolic-Numeric Computation (SNC)*, pages 161–169, 2007.

[25] John H. Reif. An $\mathcal{O}(n \log^3 n)$ algorithm for the real root problem. In *Proc. 34th Annual Symp. on Foundations of Computer Science (FOCS)*, pages 626–635, 1993.

[26] F. Rouillier and Z. Zimmermann. Efficient isolation of polynomial's real roots. *J. of Computational and Applied Mathematics*, 162(1):33–50, 2004.

[27] M. Sagraloff. On the complexity of real root isolation. *CoRR*, abs/1011.0344, 2010.

[28] A. Schönhage. The fundamental theorem of algebra in terms of computational complexity. Preliminary Report, Math. Inst. Univ. Tübingen, Germany, 1982.

[29] Arnold Schönhage. Equation solving in terms of computational complexity. In *Proc. Int. Congress of Mathematics*, volume 1, pages 131–153, Berkeley, 1986.

[30] V. Sharma. Complexity of real root isolation using continued fractions. *Theor. Comput. Sci.*, 409(2):292–310, 2008.

[31] Elias P. Tsigaridas and Ioannis Z. Emiris. Univariate polynomial real root isolation: Continued fractions revisited. In Y. Azar and T. Erlebach, editors, *Proc. 14th European Symp. of Algorithms (ESA)*, volume 4168 of *LNCS*, pages 817–828, Zurich, Switzerland, 2006. Springer Verlag.

[32] Elias P. Tsigaridas and Ioannis Z. Emiris. On the complexity of real root isolation using Continued Fractions. *Theor. Comput. Sci.*, 392:158–173, 2008.

[33] J. V. Uspensky. *Theory of Equations*. McGraw-Hill, 1948.

[34] A. van der Poorten. An introduction to continued fractions. In *Diophantine analysis*, pages 99–138. Cambridge University Press, 1986.

[35] A. J. H. Vincent. Sur la résolution des équations numériques. *J. Math. Pures Appl.*, 1:341–372, 1836.

[36] J. von zur Gathen and J. Gerhard. Fast Algorithms for Taylor Shifts and Certain Difference Equations. In *Proc. ACM ISSAC*, pages 40–47, 1997.

[37] C.K. Yap. *Fundamental Problems of Algorithmic Algebra*. Oxford University Press, New York, 2000.