Advances on the Continued Fractions Method Using Better Estimations of Positive Root Bounds

Alkiviadis G. Akritas[1], Adam W. Strzeboński[2], and Panagiotis S. Vigklas[1]

[1] University of Thessaly, Department of Computer and Communication Engineering, GR-38221 Volos, Greece
{akritas, pviglas}@uth.gr
[2] Wolfram Research, Inc., 100 Trade Center Drive, Champaign, IL 61820, USA
adams@wolfram.com

Abstract. We present an implementation of the Continued Fractions (CF) real root isolation method using a recently developed upper bound on the positive values of the roots of polynomials. Empirical results presented in this paper verify that this implementation makes the CF method *always* faster than the Vincent-Collins-Akritas bisection method[3], or any of its variants.

1 Introduction

We begin by first reviewing some basic facts about the continued fractions method for isolating the positive roots of polynomials. This method is based on Vincent's theorem of 1836, [Vincent 1836], which states:

Theorem 1. *If in a polynomial, $p(x)$, of degree n, with rational coefficients and without multiple roots we perform sequentially replacements of the form*

$$x \leftarrow \alpha_1 + \tfrac{1}{x}, x \leftarrow \alpha_2 + \tfrac{1}{x}, x \leftarrow \alpha_3 + \tfrac{1}{x}, \dots$$

where $\alpha_1 \geq 0$ is a random non negative integer and $\alpha_2, \alpha_3, \dots$ are random positive integers, $\alpha_i > 0$, $i > 1$, then the resulting polynomial either has no sign variations or it has one sign variation. In the last case the equation has exactly one positive root, which is represented by the continued fraction

$$\alpha_1 + \cfrac{1}{\alpha_2 + \cfrac{1}{\alpha_3 + \cfrac{1}{\ddots}}}$$

whereas in the first case there are no positive roots.

[3] misleadingly referred to (by several authors) as "Descarte's method"

Note that if we represent by $\frac{ax+b}{cx+d}$ the continued fraction that leads to a transformed polynomial $f(x) = (cx+d)^n p(\frac{ax+b}{cx+d})$, with one sign variation, then the single positive root of $f(x)$—in the interval $(0, \infty)$—corresponds to *that* positive root of $p(x)$ which is located in the open interval with endpoints $\frac{b}{d}$ and $\frac{a}{c}$. These endpoints are *not* ordered and are obtained from $\frac{ax+b}{cx+d}$ by replacing x with 0 and ∞, respectively. See the papers by Alesina & Galuzzi, [Alesina and Galuzzi 1998] and Chapter 7 in [Akritas 1989] for a complete historical survey of the subject and implementation details respectively[4].

Cauchy's method, for computing bounds on the positive roots of a polynomial, was mainly used until now in the Continued Fraction (CF) real root isolation method, [Akritas and Strzeboński 2005]. In the SYNAPS implementation of the CF method, [Tsigaridas and Emiris 2006], Emiris and Tsigaridas used Kioustelidis method, [Kioustelidis 1986] for computing such bounds and independently verified the results obtained in [Akritas and Strzeboński 2005].

Both implementations of the CF method showed that its "Achilles heel" was the case of very many very large roots. In this case as Rouillier and Zimmermann, also showed, [Rouillier and Zimmermann 2004], the CF method was up to 4 times slower than REL, the bisection method they proposed—a variant of the Vincent-Collins-Akritas method, [Collins and Akritas 1976]. Table 1 presented below, is an exact copy of the last table (Table 4), found in [Akritas and Strzeboński 2005].

Table 1. Products of factors (x-randomly generated integer root). All computations were done on a 850 MHz Athlon PC with 256 MB RAM; (s) stands for time in seconds and (MB) for the amount of memory used, in MBytes.

Roots (bit length)	Degree	No. of roots	CF T (s)/M (MB)	REL T (s)/M (MB)
10	100	100	0.8/1.82	0.61/1.92
10	200	200	2.45/2.07	10.1/2.64
10	500	500	33.9/3.34	878/8.4
1000	20	20	0.12/1.88	0.044/1.83
1000	50	50	16.7/3.18	4.27/2.86
1000	100	100	550/8.9	133/6.49

The last three lines of Table 1 demonstrate the weaker performance of CF in the case of very many, very large roots. However, we recently generalized and extended a theorem by Ştefănescu, [Ştefănescu 2005], and developed a new method for computing upper bounds on the positive roots of polynomials, [Akritas, Strzeboński & Vigklas 2006]. As was verified, this method provides the sharpest upper bounds on the positive roots of polynomials. In this paper, we incorporated into CF this new method for computing upper bounds for positive roots. It turns out that with this modifica-

[4] Alesina and Galuzzi point out in their work that Vincent's theorem can be implemented in various ways; the Vincent-Collins-Akritas bisection method is also one such implementation.

tion, the CF algorithm is now *always* faster than that of Vincent-Collins-Akritas, or any of its variants.

2 Algorithmic Background

In this section we present the CF algorithm (where we correct a misprint in Step 5 that appeared in [Akritas and Strzeboński 2005] and explain where the new bound on the positive roots is used.

2.1 Description of the Continued Fractions Algorithm CF

Using the notation of the paper [Akritas and Strzeboński 2005], let $f \in Z[x] \setminus \{0\}$. By $sgc(f)$ we denote the number of sign changes in the sequence of nonzero coefficients of f. For nonnegative integers a, b, c, and d, such that $ad - bc \neq 0$, we put

$$intrv(a, b, c, d) := \Phi_{a,b,c,d}((0, \infty))$$

where

$$\Phi_{a,b,c,d} : (0, \infty) \ni x \longrightarrow \frac{ax + b}{cx + d} \in (min(\frac{a}{c}, \frac{b}{d}), max(\frac{a}{c}, \frac{b}{d}))$$

and by *interval data* we denote a list

$$\{a, b, c, d, p, s\}$$

where p is a polynomial such that the roots of f in $intrv(a, b, c, d)$ are images of positive roots of p through $\Phi_{a,b,c,d}$, and $s = sgc(p)$.

The value of parameter α_0 used in step 4 below needs to be chosen empirically. In our implementation $\alpha_0 = 16$.

Algortihm Continued Fractions (CF).
Input: A squarefree polynomial $f \in Z[x] \setminus \{0\}$
Output: The list *rootlist* of positive roots of f.

1. Set *rootlist* to an empty list. Compute $s \leftarrow sgc(f)$. If $s = 0$ return an empty list. If $s = 1$ return $\{(0, \infty)\}$. Put interval data $\{1, 0, 0, 1, f, s\}$ on *intervalstack* .
2. If *intervalstack* is empty, return *rootlist*, else take interval data $\{a, b, c, d, p, s\}$ off *intervalstack*.
3. Compute a lower bound α on the positive roots of p.
4. If $\alpha > \alpha_0$ set $p(x) \leftarrow p(\alpha x)$, $a \leftarrow \alpha a$, $c \leftarrow \alpha c$, and $\alpha \leftarrow 1$.
5. If $\alpha \geq 1$, set $p(x) \leftarrow p(x + \alpha)$, $b \leftarrow \alpha a + b$, and $d \leftarrow \alpha c + d$. If $p(0) = 0$, add $[b/d, b/d]$ to *rootlist*, and set $p(x) \leftarrow p(x)/x$. Compute $s \leftarrow sgc(p)$. If $s = 0$ go to step 2. If $s = 1$ add $intrv(a, b, c, d)$ to *rootlist* and go to step 2.
6. Compute $p_1(x) \leftarrow p(x+1)$, and set $a_1 \leftarrow a$, $b_1 \leftarrow a+b$, $c_1 \leftarrow c$, $d_1 \leftarrow c+d$, and $r \leftarrow 0$. If $p_1(0) = 0$, add $[b_1/d_1, b_1/d_1]$ to *rootlist*, and set $p_1(x) \leftarrow p_1(x)/x$, and $r \leftarrow 1$. Compute $s_1 \leftarrow sgc(p_1)$, and set $s_2 \leftarrow s - s_1 - r$, $a_2 \leftarrow b$, $b_2 \leftarrow a + b$, $c_2 \leftarrow d$, and $d_2 \leftarrow c + d$.

7. If $s_2 > 1$, compute $p_2(x) \leftarrow (x+1)^m p(\frac{1}{x+1})$, where m is the degree of p. If $p_2(0) = 0$, set $p_2(x) \leftarrow p_2(x)/x$. Compute $s_2 \leftarrow sgc(p_2)$.
8. If $s_1 < s_2$, swap $\{a_1, b_1, c_1, d_1, p_1, s_1\}$ with $\{a_2, b_2, c_2, d_2, p_2, s_2\}$.
9. If $s_1 = 0$ goto step 2. If $s_1 = 1$ add $intrv(a_1, b_1, c_1, d_1)$ to *rootlist*, else put interval data $\{a_1, b_1, c_1, d_1, p_1, s_1\}$ on *intervalstack*.
10. If $s_2 = 0$ goto step 2. If $s_2 = 1$ add $intrv(a_2, b_2, c_2, d_2)$ to *rootlist*, else put interval data $\{a_2, b_2, c_2, d_2, p_2, s_2\}$on *intervalstack*. Go to step 2.

Please note that the lower bound, α, on the positive roots of $p(x)$ is computed in Step 3, and used in Step 5.

To compute this bound we generalized Ştefănescu's theorem, [Ştefănescu 2005], in the sense that Theorem 2 (see below) applies to polynomials with any number of sign variations; moreover we have introduced the concept of *breaking up* a positive coefficient into several parts to be paired with negative coefficients (of lower order terms).

Theorem 2. *Let* $p(x)$

$$p(x) = \alpha_n x^n + \alpha_{n-1} x^{n-1} + \ldots + \alpha_0, \quad (\alpha_n > 0) \tag{1}$$

be a polynomial with real coefficients and let $d(p)$ *and* $t(p)$ *denote the degree and the number of its terms, respectively.*

Moreover, assume that $p(x)$ *can be written as*

$$p(x) = q_1(x) - q_2(x) + q_3(x) - q_4(x) + \ldots + q_{2m-1}(x) - q_{2m}(x) + g(x), \tag{2}$$

where all the polynomials $q_i(x)$, $i = 1, 2, \ldots, 2m$ *and* $g(x)$ *have only positive coefficients. In addition, assume that for* $i = 1, 2, \ldots, m$ *we have*

$$q_{2i-1}(x) = c_{2i-1,1} x^{e_{2i-1,1}} + \ldots + c_{2i-1,t(q_{2i-1})} x^{e_{2i-1,t(q_{2i-1})}}$$

and

$$q_{2i}(x) = b_{2i,1} x^{e_{2i,1}} + \ldots + b_{2i,t(q_{2i})} x^{e_{2i,t(q_{2i})}},$$

where $e_{2i-1,1} = d(q_{2i-1})$ *and* $e_{2i,1} = d(q_{2i})$ *and the exponent of each term in* $q_{2i-1}(x)$ *is greater than the exponent of each term in* $q_{2i}(x)$. *If for all indices* $i = 1, 2, \ldots, m$, *we have*

$$t(q_{2i-1}) \geq t(q_{2i}),$$

then an upper bound of the values of the positive roots of $p(x)$ *is given by*

$$ub = \max_{\{i=1,2,\ldots,m\}} \left\{ \left(\frac{b_{2i,1}}{c_{2i-1,1}} \right)^{\frac{1}{e_{2i-1,1} - e_{2i,1}}}, \ldots, \left(\frac{b_{2i,t(q_{2i})}}{c_{2i-1,t(q_{2i})}} \right)^{\frac{1}{e_{2i-1,t(q_{2i})} - e_{2i,t(q_{2i})}}} \right\},$$

for any permutation of the positive coefficients $c_{2i-1,j}$, $j = 1, 2, \ldots, t(q_{2i-1})$. *Otherwise, for each of the indices* i *for which we have*

$$t(q_{2i-1}) < t(q_{2i}),$$

we ***break up*** *one of the coefficients of* $q_{2i-1}(x)$ *into* $t(q_{2i}) - t(q_{2i-1}) + 1$ *parts, so that now* $t(q_{2i}) = t(q_{2i-1})$ *and apply the same formula (3) given above.*

For a proof of this theorem see [Akritas, Strzeboński & Vigklas 2006]. It turns out that all existing methods (i.e. Cauchy's, Lagrange-McLaurent, Kioustelidis's, etc) for computing upper bounds on the positive roots of a polynomial, are special cases of Theorem 2.

In this recent paper of ours, we also presented two new implementations of Theorem 2, the combination of which yields the best upper bound on the positive roots of a polynomial. These implementation are:

(a) "*first*–λ" implementation of Theorem 2. For a polynomial $p(x)$, as in (2), with λ negative coefficients we first take care of all cases for which $t(q_{2i}) > t(q_{2i-1})$, by breaking up the last coefficient $c_{2i-1,t(q_{2i})}$, of $q_{2i-1}(x)$, into $t(q_{2i}) - t(q_{2i-1}) + 1$ *equal* parts. We then pair each of the first λ positive coefficients of $p(x)$, encountered as we move in non-increasing order of exponents, with the first unmatched negative coefficient.

(b) "*local-max*" implementation of Theorem 2. For a polynomial $p(x)$, as in (1), the coefficient $-\alpha_k$ of the term $-\alpha_k x^k$ in $p(x)$ —as given in Eq. (1)— is paired with the coefficient $\frac{\alpha_m}{2^t}$, of the term $\alpha_m x^m$, where α_m is the largest positive coefficient with $n \geq m > k$ and t indicates the number of times the coefficient α_m has been used.

As an upper bound on the positive roots of a polynomial we take the minimum of the two bounds produced by implementations (a) & (b), mentioned above. This minimum of the two bounds is first computed in Step 3 and then used in Step 5 of CF.

3 Empirical Results

Below we recalculate the results of Table 1, comparing the timings in seconds (s) for: (a) the CF using Cauchy's rule (CF_OLD), (b) the CF using the new rule for computing upper bounds (CF_NEW), and (c) REL.

Due to the different computational environment the times differ substantially, but they confirm the fact that now the CF is always faster.

Table 2. Products of terms $x - r$ with random integer r. The tests were run on a laptop computer with 1.8 Ghz Pentium M processor, running a Linux virtual machine with 1.78 GB of RAM.

Bit len of rts	Deg	CF_OLD Time(s) Average (Min/Max)	CF_NEW Time(s) Average (Min/Max)	REL Average (Min/Max)	Memory (MB) CFO/CFN/REL
10	100	0.314 (0.248/0.392)	0.253 (0.228/0.280)	0.346 (0.308/0.384)	4.46/4.48/4.56
10	200	1.74 (1.42/2.33)	1.51 (1.34/1.66)	3.90 (3.72/4.05)	4.73/4.77/5.35
10	500	17.6 (16.9/18/7)	17.4 (16.3/18.1)	129 (122/140)	6.28/6.54/11.8
1000	20	0.066 (0.040/0.084)	0.031 (0.024/0.040)	0.038 (0.028/0.044)	4.57/4.62/4.51
1000	50	1.96 (1.45/2.44)	0.633 (0.512/0.840)	1.03 (0.916/1.27)	5.87/6.50/5.55
1000	100	52.3 (36.7/81.3)	12.7 (11.3/14.6)	17.2 (16.1/18.7)	10.4/11.7/9.17

Again, of interest are the last three lines of Table 2, where as in Table 1 the performance of CF_OLD is worst than REL—at worst 3 times slower as the last entry indicates. However, from these same lines of Table 2 we observe that CF_NEW is now always faster than REL—at best twice as fast, as seen in the 5-th line.

4 Conclusions

In this paper we have examined the behavior of CF on the special class of polynomials with very many, very large roots—a case where CF exhibited a certain weakness. We have demonstrated that, using our recently developed rule for computing upper bounds on the positive roots of polynomials, CF is speeded up by a considerable factor and is now always faster than any other real root isolation method.

References

[Akritas 1978] Akritas, A.:"Vincent's theorem in algebraic manipulation"; Ph.D. Thesis, North Carolina State University, Raleigh, NC, (1978).

[Akritas 1982] Akritas, A.:"Reflections on a pair of theorems by Budan and Fourier"; Mathematics Magazine, 55, 5, (1982), 292–298.

[Akritas 1989] Akritas, A.:"Elements of Computer Algebra with Applications"; John Wiley Interscience, New York, (1989).

[Akritas and Strzeboński 2005] Akritas, A., Strzebonski, A.:"A comparative study of two real root isolation methods"; Nonlinear Analysis: Modelling and Control, 10, 4, (2005), 297-304.

[Akritas, Strzeboński & Vigklas 2006] Akritas, A., Strzebonski, A., Vigklas, P.:"Implementations of a New Theorem for Computing Bounds for Positive Roots of Polynomials"; Computing, 78, (2006), 355–367.

[Alesina and Galuzzi 1998] Alesina, A., Galuzzi, M.: "A new proof of Vincent's theorem"; L'Enseignement Mathemathique, 44, (1998), 219–256.

[Collins and Akritas 1976] Collins, E. G., Akritas, G. A.: "Polynomial real root isolation using Descartes' rule of signs"; Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computations, Yorktown Heights, N.Y., (1976), 272–275.

[Kioustelidis 1986] Kioustelidis, B.:"Bounds for positive roots of polynomials"; J. Comput. Appl. Math., 16, 2, (1986), 241–244.

[Rouillier and Zimmermann 2004] Rouillier, F., Zimmermann, P.: "Efficient isolation of polynomial's real roots"; Journal of Computational and Applied Mathematics, 162, (2004), 33–50.

[Ştefănescu 2005] Ştefănescu, D.:"New bounds for positive roots of polynomials"; Journal of Universal Computer Science, 11, 12, (2005), 2132-2141.

[Tsigaridas and Emiris 2006] Tsigaridas, P. E., Emiris, Z. I.: "Univariate polynomial real root isolation: Continued fractions revisited"; (Y. Azar and T. Erlebach (Eds.)), ESA 2006, LNCS 4168, (2006), 817–828.

[Vincent 1836] Vincent, A.J.H.:"Sur la resolution des équations numériques"; Journal de Mathématiques Pures et Appliquées, 1, (1836), 341–372.