Journal of Symbolic Computation ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Symbolic Computation

www.elsevier.com/locate/jsc

Near optimal subdivision algorithms for real root isolation

Prashant Batra [a], Vikram Sharma [b]

[a] Technische Universität Hamburg-Harburg, Hamburg, Germany
[b] Institute of Mathematical Sciences, Chennai, 600113, India

A R T I C L E   I N F O

Article history:
Received 2 September 2015
Accepted 3 September 2016
Available online xxxx

Keywords:
Real root isolation
Subdivision algorithms
Newton diagram
Continuous amortization
Integral analysis

A B S T R A C T

Isolating real roots of a square-free polynomial in a given interval is a fundamental problem in computational algebra. Subdivision based algorithms are a standard approach to solve this problem. For instance, Sturm's method, or various algorithms based on the Descartes's rule of signs. For the benchmark problem of isolating all the real roots of a polynomial of degree n and root separation σ, the size of the subdivision tree of most of these algorithms is bounded by $O(\log 1/\sigma)$ (assume $\sigma < 1$). Moreover, it is known that this is optimal for subdivision algorithms that perform uniform subdivision, i.e., the width of the interval decreases by some constant. Recently Sagraloff (2012) and Sagraloff–Mehlhorn (2016) have developed algorithms for real root isolation that combine subdivision with Newton iteration to reduce the size of the subdivision tree to $O(n(\log(n \log 1/\sigma)))$.

We describe a subroutine that reduces the size of the subdivision tree of any subdivision algorithm for real root isolation. The subdivision tree size of our algorithm using predicates based on either the Descartes's rule of signs or Sturm sequences is bounded by $O(n \log n)$, which is close to the optimal value of $O(n)$. The corresponding bound for the algorithm EVAL, which uses certain interval-arithmetic based predicates, is $O(n^2 \log n)$. Our analysis differs in two key aspects from earlier approaches. First, we use the general technique of continuous amortization from Burr–Krahmer–Yap (2009), and extend it to handle non-uniform subdivisions; second, we use the geometry of clusters of roots instead of root bounds. The latter aspect enables us to derive a bound on the

E-mail addresses: batra@tuhh.de (P. Batra), vikram@imsc.res.in (V. Sharma).

http://dx.doi.org/10.1016/j.jsc.2016.11.004

subdivision tree that is independent of the root separation σ. The number of Newton iterations is bounded by $O(n \log\log(1/\sigma))$.

1. Introduction

Given a square-free polynomial $f \in \mathbb{R}[x]$ of degree n, the **real root isolation** problem is to isolate the real roots of f in an input interval I_0, i.e., to compute disjoint intervals which contain exactly one real root of f, and together contain all roots of f in I_0. Subdivision based algorithms have been successful in addressing the problem (see Collins and Akritas, 1976; Krandick, 1995; Johnson, 1998; Rouillier and Zimmermann, 2003). For non-subdivision based approaches that are nearly optimal and handle the more general problem of complex root isolation see Schönhage (1982), Pan (1996), Neff and Reif (1996), Pan (2002), and Mehlhorn et al. (2015).

A general subdivision algorithm uses two predicates, given an interval I: an exclusion predicate $C_0(I)$, which if true means I has no roots; an inclusion predicate $C_1(I)$, which if true means I has exactly one root. The algorithm outputs a **root-partition** $\mathcal{P}$ of I_0, i.e., a set of pairwise disjoint open intervals such that for each interval either C_0 or C_1 holds, and $I_0 \setminus \mathcal{P}$ contains no roots of f. To compute isolating intervals for roots of f, we can check the sign of f at the endpoints of the intervals in $\mathcal{P}$. The following generic subdivision algorithm constructs a root-partition:

```
Isolate(f, I_0)
0. Preprocessing step.
1. Initialize a queue Q with I_0, and P ← ∅.
2. While Q is not empty
      Remove an interval I = (a, b) from Q.
      If C_0(I) ∨ C_1(I) then add I to P.
      else   ◃ Subdivide I
         Let m ← (a + b)/2.
         Push (a, m) and (m, b) into Q.
3. Output P.
```

The algorithm is guaranteed to terminate for square-free polynomials; otherwise we get an infinite sequence of intervals converging to a multiple root. Some standard choices of the predicates and the corresponding algorithms are:

(1) Sturm sequences and Sturm's method (see Davenport, 1985),
(2) Descartes's rule of signs and the Descartes method (see Collins and Akritas, 1976),
(3) Interval-arithmetic based approaches and `Eval` (see Burr et al., 2009).

One measure of the complexity of a subdivision algorithm is the size of the subdivision tree constructed by the algorithm for the benchmark problem of isolating all the real roots of a square-free polynomial. We express the bounds in terms of n and the root separation σ of f (assume $\sigma < 1$). For the first two algorithms mentioned above a bound of $O(\log 1/\sigma)$ is known (Davenport, 1985 and Eigenwillig et al., 2006, respectively). For `Eval`, Sharma and Yap (2012) derive a weaker bound of $O(n^2 + \log 1/\sigma)$. Moreover, Eigenwillig et al. (2006) give examples of polynomials for whom the size of the subdivision tree is $\Omega(\log 1/\sigma)$ for any algorithm doing uniform subdivision, i.e., the width of the interval decreases by a constant (in most cases, by half) at each subdivision step. Uniform subdivision cannot improve on this bound because it only gives linear convergence to a "root cluster", i.e., a set of roots which are relatively closer to each other than to any other root. In this scenario it is natural to use the Newton iteration to improve the convergence