Solving Decidability Problems with Interval Arithmetic*

German Tischler
Lehrstuhl für Informatik II, Universität Würzburg, am Hubland, 97074 Würzburg, Germany
tischler@informatik.uni-wuerzburg.de

Jürgen Wolff von Gudenberg
Lehrstuhl für Informatik II, Universität Würzburg, am Hubland, 97074 Würzburg, Germany
wolff@informatik.uni-wuerzburg.de

Abstract

An affine iterated function system (IFS, [4]) over the reals is given by a set of contractive affine transformations on a complete metric space $\mathbb{R}^k$ for some positive natural number k.

Thus an interesting problem is to decide, whether a given matrix M describes a contraction mapping, i.e. whether there exists some $s < 1$ such that $|Mz| \leq s|z|$ for each $z \in \mathbb{R}^k$, where $|\cdot|$ denotes the Euclidean norm. Using the characteristic polynomial and interval methods we present a simple algorithm deciding the contraction mapping property for square matrices over the rational numbers. Furthermore, we show that interval arithmetic can be used to enclose the roots of complex polynomials with coefficients having rational real and imaginary parts at arbitrary precision such that each computed interval can be annotated with the exact number of contained roots.

Keywords: interval computations, polynomials, roots, decidability
AMS subject classifications: 26C10, 03B25

1 Preliminaries

Let the sum and product of two subsets A and B of the complex numbers be given by $A + B = \{a + b | a \in A, b \in B\}$ and $AB = \{ab | a \in A, b \in B\}$ respectively. We identify the number c with the set $\{c\}$ for each $c \in \mathbb{C}$. For three non-empty subsets A, B and C of $\mathbb{C}$, the equation $A(B+C) = AB+AC$ in general does not hold (e.g. let $A = [-1, 2]$,

*Submitted: January 11, 2009; Revised: February 10, 2010; Accepted: March 1, 2010.

$B = [-5, 2]$ and $C = [-4, 3]$, then $A(B + C) = [-18, 10] \neq [-18, 11] = AB + AC$). To see why, observe that

$$\begin{aligned} A(B+C) &= \{a(b+c) | a \in A, b \in B, c \in C\} \\ &= \{ab + ac | a \in A, b \in B, c \in C\} \ . \end{aligned} \tag{1}$$

However, we can write

$$\begin{aligned} \{ab + ac | a \in A, b \in B, c \in C\} &\subseteq \{a_1 b + a_2 c | a_1, a_2 \in A, b \in B, c \in C\} \\ &= \{ab | a \in A, b \in B\} + \{ac | a \in A, c \in C\} \\ &= AB + AC \ . \end{aligned} \tag{2}$$

Assume that the function $p : \mathbb{C} \mapsto \wp(\mathbb{C})$ (where $\wp(\mathbb{C})$ denotes the power set of $\mathbb{C}$) is given by

$$p(x) = (x - S_1)(x - S_2) \ldots (x - S_n) = \prod_{j=1}^{n} (x - S_j) \tag{3}$$

with $S_j \subseteq \mathbb{C}, S_j \neq \emptyset$ for $j = 1, \ldots, n$. Then it is clear from above that there is a function $P : \mathbb{C} \mapsto \wp(\mathbb{C})$ given by $P(x) = \sum_{j=0}^{n} S'_j x^j$ such that

1. $p(x) \subseteq P(x)$ for each $x \in \mathbb{C}$ and
2. each S'_j for $j = 0, \ldots, n$ is the result of the evaluation of an expression consisting of the sets $S_j, j = 1, \ldots, n$ and the operations sum and product as defined above.

Such a function P can be obtained using the following steps:

1. Completely expand the expression given in equation 3. This produces an expression of the form
$$P''(x) = \sum_{j=1}^{k} \prod_{i=1}^{n} S_i^{q_{j,i}} x^{r_j} \tag{4}$$
for some natural number k, $q_{j,i} \in \{0, 1\}$ for $1 \leq j \leq k, 1 \leq i \leq n$ and $0 \leq r_j \leq n$ for $1 \leq j \leq k$. According to the discussion above, $P''(x) \supseteq p(x)$ holds for each $x \in \mathbb{C}$.
2. Group the summands according to the respective power of x. This produces the expression
$$P'(x) = \sum_{j=0}^{n} x^j \left(\sum_{i=1}^{k_j} \prod_{l=1}^{n} S_l^{q_{i,l}} \right) \tag{5}$$
such that k_j is a natural number for each $0 \leq j \leq n$ and $q_{i,l} \in \{0, 1\}$ for each $1 \leq i \leq k_j$ and $1 \leq l \leq n$. As we have only factored out singletons x^i for $0 \leq i \leq n$ during the transformation, we have $P'(x) = P''(x)$ for each $x \in \mathbb{C}$.
3. Evaluate the coefficient expressions for each power of x. This finally produces the expression
$$P(x) = \sum_{j=0}^{n} S'_j x^j, \tag{6}$$
where $S'_j \subseteq \mathbb{C}, S'_j \neq \emptyset$ for $0 \leq j \leq n$ and $P(x) = P'(x)$ for each $x \in \mathbb{C}$. Note that if for some reason, e.g. due to the usage of some kind of interval arithmetic, we are only able to compute supersets of the sets S'_j instead of the real sets, then this is no problem as then we obtain $P(x) \supseteq P'(x)$ for each $x \in \mathbb{C}$, which still ensures that $P(x) \supseteq p(x)$ for each $x \in \mathbb{C}$.

2 The Problem

Let M denote a square matrix over the real numbers. M describes a contraction mapping, iff there exists some $s < 1$ such that $|Mz| \leq s|z|$ for all real vectors z of the appropriate dimension. This property holds, iff $\max_{z \neq 0} |Mz|/|z| = |M|_2 \leq s < 1$, where $|M|_2$ denotes the spectral norm of M. $|M|_2$ equals the square root of the maximal eigenvalue of the real symmetric positive-semidefinite matrix $S = M^T M$ (where M^T denotes the transpose of M). These properties of S guarantee that the characteristic polynomial χ_S of S, whose roots are the eigenvalues of S, has only real roots. M is a contraction mapping iff the largest such root is smaller than 1. Thus to decide, whether M is contraction, we can use any algorithm which decides if the largest eigenvalue of S is smaller than 1. One of the algorithms we present in this paper is such a decision algorithm. The other presented algorithms are generalizations of the approach we describe.

The algorithms we present in this paper are not answers to the general question, whether the treated problems are decidable at all. The well-known Tarski-Seidenberg theorem can be used to solve these (and even much more general) questions regarding properties of sets of real numbers (cf. [5]). To our best knowledge the approach using interval arithmetic we present in this paper has not been proposed before in this form. It provides solutions which are very simple to describe and to implement. Thus for some settings, they may be a good choice for applications.

In the following we consider polynomials that only have real roots. As we are discussing decidability issues and want to avoid undecidability of basic properties like the order between single numbers, we consider only polynomials with rational coefficients. Thus the respective roots of the polynomials are algebraic numbers. Let

$$p(x) = \sum_{i=0}^{n} c_i x^i \tag{7}$$

denote a polynomial of degree n. Assume that p only has single roots and that we have computed a set of n disjoint real intervals $I_1, \ldots, I_n$ and have proved for each interval that it contains a root of p using the intermediate value theorem. It is easy to see that this is possible using bisection. Then we know that the real interval coefficient polynomial $P : \mathbb{R} \mapsto \wp(\mathbb{R})$ given by

$$\prod_{j=1}^{n} (x - I_j) \subseteq \sum_{j=0}^{n} C_j x^j = P(X) \tag{8}$$

satisfies the constraints

1. $p(x) \in P(X)$ for each real interval X and $x \in X$ and
2. $c_j \in C_j$ for each $j = 0, \ldots, n$.

If p has multiple roots, then the situation is only slightly more complicated. We can use interval bisection to compute a set of intervals $I_1, \ldots, I_m$ such that $0 \in p(I_j)$ for each $j = 1, \ldots, m$ and the union of the I_j contains all roots of p. Now each such interval I_j possibly contains a root. There is, however, at least one choice $t = (i_1, i_2, \ldots, i_m) \in \mathbb{N}^m$ such that

1. $\sum_{j=1}^{m} i_j = n$ and
2. $\prod_{j=1}^{m} (x - I_j)^{i_j} \subseteq \sum_{j=0}^{n} C_j x^j$ such that $c_j \in C_j$ for $j = 1, \ldots, n$.

If an m-tuple t satisfies these constraints for the intervals $I_1, \ldots, I_m$, then we say that $(t, I_1, \ldots, I_m)$ generates p. If there exists some m-tuple t for the intervals $I_1, \ldots, I_m$ such that $(t, I_1, \ldots, I_m)$ generates p, then we say that the set $\{I_1, \ldots, I_m\}$ is able to generate p.

Our goal is to give a simple interval arithmetic based algorithm deciding whether all roots of a polynomial having only real roots are less than a given rational constant d. We may imagine that the presented algorithm observes two processes running in parallel. The first process tries to prove that all roots of the given polynomial are smaller than d and terminates as soon as it is able give a proof. The second process tries to prove that there is at least one root that is not smaller than d, by first checking whether d is a root and otherwise by enclosing all roots below d at ever increasing precision, i.e. with decreasing width of the intervals, and proving that these roots are at some precision no longer able to generate the complete polynomial. If it finds a proof than it terminates.

We are sure that one of the two processes will eventually terminate. As soon as one of the two has terminated, the decision problem is solved. The algorithm uses subdivision and is described in more detail in section 4.

3 The Subdivision Process

3.1 Real Interval Bisection

If we apply interval bisection to our starting interval $[-b, b]$, where b is a root bound for the polynomial, we can construct sets of intervals $I_1, \ldots, I_m$ such that $0 \in p(I_j)$ and the diameter of each interval I_j is at most 2^{-k} for each $k \in \mathbb{N}$.

3.2 Interval Newton

The interval Newton method (cf. [2, 1, 3]) for real functions is a generalization of the non-interval Newton method for real functions. It computes enclosures of all real roots of a function in a given start interval using division by intervals containing zero. Hence different ways to extend interval division have been suggested. Kulisch [9] defines an extended division by an interval containing zero by returning 2 unbounded intervals. If the numerator is a proper interval, there is a gap between the upper bound of the lower interval and the lower bound of the upper interval. This gap is important for the interval Newton method, because it certainly does not contain any zero of the function. The interval hull of these two intervals is the complete real line. Hence a simple extension of interval arithmetic delivers that as a result of extended division. This kind of extended interval arithmetic was introduced by Walster in [10].

Real Newton

The Newton iteration for a real function $f(x)$ with first derivative $f'(x) = \frac{df}{dx}(x)$ is given by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \quad . \tag{9}$$

In the example of $f(x) = x^2 - 2$, $f'(x) = 2x$ and $x_0 = 2$, we obtain a sequence x_k such that $\lim_{k\to\infty} x_k = \sqrt{2}$. There are however some cases where the method fails, e.g. if we reach some x_k such that $f'(x_k) = 0$.

Interval Newton

Let $f(x)$ be a continuous real function and $f'(x) = \frac{df}{dx}(x)$. The interval Newton method is derived from the Newton method. Assume that the equation

$$f(z) \in f(y) - f'(X)(z - y) \tag{10}$$

holds for the real interval X and $y, z \in X$. As we are searching roots, we assume $f(z) = 0$, i.e.

$$0 \in f(y) - f'(X)(z - y) \tag{11}$$

which is equivalent to

$$z \in y - \frac{f(y)}{f'(X)} \tag{12}$$

for y, z in the real interval X. One iteration of the interval Newton method is given by

$$\bigcup_j X_{(k+1)_j} = \bigcup_j \left(X_{k_j} \cap \left(y_{k_j} - \frac{f(y_{k_j})}{f'(X_{k_j})} \right) \right) \tag{13}$$

where y_{k_j} is usually chosen as the middle of the interval X_{k_j} (or a suitable alternative in X_{k_j}, if X_{k_j} is not a finite interval). If the interval evaluation of the derivative f' of f contains the number 0 for X_{k_j}, then the division may produce two separate result intervals, which is why we use the union operators on the left and right side of equation 13. It is a well known result in interval arithmetic that for each given finite set of start intervals $\{X_{0_0}, \ldots, X_{0_k}\}$ and each given real function f, the method converges to a set of intervals $\{Y_0, \ldots, Y_l\}$ such that all real roots of f in each interval $X_{0,i}$ for $i = 0, \ldots, k$ are contained in the union of the intervals $Y_0, \ldots, Y_l$ and that the intersection used on the right side of equation 13 does not remove any potential roots of f from the computed intervals.

The following example shows that the the interval Newton method may get stuck in some situations. If this happens, we can enforce progress by substituting the corresponding step of the interval Newton method by one step of interval bisection.

Example 1 *Let* $f(x) = (x-2)^2$. *Then* $f'(x) = 2x - 4$. *Let* $X_{0_0} = [0, 4]$ *and* $X_{0_j} = \emptyset$ *for* $j \neq 0$. *Then the Newton iteration yields*

$$\begin{array}{rcl} \bigcup_j X_{1_j} & = & [0,4] \cap \left(2 - \frac{(2-2)^2}{2[0,4]-4}\right) \\ & = & [0,4] \cap \left(2 - \frac{0}{[-4,4]}\right) \\ & = & [0,4] \cap (2 - [-\infty, \infty]) \\ & = & [0,4] \cap [-\infty, \infty] \\ & = & [0,4], \end{array} \tag{14}$$

i.e. the step has not lead to any progress.

The results of this example change significantly, if we use Kulisch's definition of extended interval arithmetic. Here we choose $\frac{0}{B} = 0 \cdot \frac{1}{(-\infty,\infty)} = 0$. Then, Newton iteration yields

$$\begin{array}{rcl} \bigcup_j X_{1_j} & = & [0,4] \cap \left(2 - \frac{(2-2)^2}{2[0,4]-4}\right) \\ & = & [0,4] \cap \left(2 - \frac{0}{[-4,4]}\right) \\ & = & [0,4] \cap (2 - 0 \cdot (-\infty, \infty)) \\ & = & [0,4] \cap [2,2] \\ & = & [2,2], \end{array} \tag{15}$$

This time the exactly representable zero was found. It is, however, not recommended to start Newton iteration with a zero of a function, since information may be lost.

The interval Newton method augmented with bisection where necessary usually converges faster to the real roots of a polynomial than pure bisection.

Existence

The existence of a zero can be proved by the intermediate value theorem. Another way to prove the existence of some roots is Brouwer's fixed point theorem.

Theorem 1 *(Fixed point theorem of Brouwer) Let $d \in \mathbb{N}, d > 0$. Every continuous function from the closed unit ball in $\mathbb{R}^d$ into itself has a fixed point.*

If on the right side of equation 13 a finite interval is mapped to a sub-interval of itself (before the application of the intersection), then this interval contains a fixed point of the iteration. As the fixed points of the iteration are the roots of the function, this implies that a root is present in the interval.

However, there are cases where neither the fixed point theorem of Brouwer nor the intermediate value theorem are able to prove the existence of a root in an interval that appears during the application of the interval Newton method.

Example 2 *Let $f(x) = (x^2 - 2)^2 = x^4 - 4x^2 + 4 = (x - \sqrt{2})^2(x + \sqrt{2})^2$. Then $f'(x) = 4x^3 - 8x$. The function f has the double roots $x = \pm\sqrt{2}$. We cannot prove any of the two roots using bisection and the intermediate value theorem, if we are using intervals that have rational lower and upper bounds. We consider the starting interval $A = [-4, 4]$. Then a step of the interval Newton method produces the successor intervals $[-\infty, -0.0138^{\omega}]$ and $[0.0138^{\omega}, \infty]$ before the intersection. (Here 8^{ω} means infinitely many repetitions of the digit 8.) Although both intervals contain a root of f, both obtained intervals are neither finite nor subsets of A.*

4 The Algorithm

The presented algorithm is inspired by the following well known theorem.

Theorem 2 *Let A be a set such that equality is decidable for the elements of the set $A \cup \overline{A}$. Then it is decidable, whether $a \in A$, if and only if both A and its complement $\overline{A}$ are recursively enumerable.*

Theorem 3 *Let $p(x) = \sum_{i=0}^{n} c_i x^i$ denote a real polynomial where all roots of p are real numbers and $c_i \in \mathbb{Q}$ for $i = 0, \ldots, n$. Further let $d \in \mathbb{Q}$. Then Algorithm 1 is able to decide, whether all roots of p are smaller than d.*

Proof: As p has only rational coefficients, all roots of p are algebraic numbers, for which order is decidable (cf. [6]). We will use interval arithmetic using rational lower and upper bounds. Thus all basic (arithmetic) operations and decidability problems (e.g. order) can be solved effectively. If p is constant, then p has either no roots and thus all roots are smaller than d or every real number x is a root of $p \equiv 0$, implying that p has roots that are at least d. Now assume that p is not constant. As d is rational, we can determine whether d is a root of p by simply evaluating p for d. If

$p(d) = 0$, then there is at least one root of p that is not smaller than d. Now assume that $p(d) \neq 0$ and $p(x)$ is not constant. Let b a positive rational root bound for p (cf. [8]). The key idea is to divide the interval $[-b, b]$ into 2 regions L_k and R_k. Let $L_0 = \{[-b, d]\}$ and $R_0 = \{[d, b]\}$. We can use interval bisection to refine the sets L_0 and R_0 to $L_k = \{\mathcal{L}_{k_1}, \ldots, \mathcal{L}_{k_{s_k}}\}$ and $R_k = \{\mathcal{R}_{k_1}, \ldots, \mathcal{R}_{k_{t_k}}\}$ such that L_k and R_k contain only intervals of diameter smaller than 2^{-k} and such that the interval evaluation of p contains zero for each $\mathcal{L}_j \in L_k$ and each $\mathcal{R}_j \in R_k$. We know that for each k the union of the intervals in L_k and R_k is able to generate an interval polynomial that satisfies the constraints given above. If all roots of p are smaller than d, then this is also true for any k and the set L_k. If on the other hand at least one root of p is not smaller than d, then there is some $k \in \mathbb{N}$ such that L_k is no longer able to generate p. Thus the decision algorithm can be formulated as shown in Algorithm 1. □

Although theoretically correct, Algorithm 1 is, at least in the presented form, usable only for very limited examples in practice. This is due to the *for each* loop starting in line 24. We can however reach a speed-up by further investigating which tuples are relevant and which are not.

Assume that we found a set of intervals $L_k = \{\mathcal{L}_{k_1}, \ldots, \mathcal{L}_{k_{m_k}}\}$ and an m_k-tuple t_k such that $(t_k, \mathcal{L}_{k_1}, \ldots, \mathcal{L}_{k_{m_k}})$ generates a polynomial p in the k-th run of the while loop starting in line 13. New intervals are constructed from the elements of L_k at the end of the loop in line 35. Assume that the interval $\mathcal{L}_{k_j}$ is split into the intervals $\mathcal{L}_{k+1_{2j+1}}$ and $\mathcal{L}_{k+1_{2j+2}}$. This implies that we have possible successor tuples such that the sum of the number of roots assigned to $\mathcal{L}_{k+1_{2j+1}}$ and $\mathcal{L}_{k+1_{2j+2}}$ equals the number of roots assigned to $\mathcal{L}_{k_j}$ by t_k. Apart from the primitive tuple (n) that is used in the first run of the loop, all relevant tuples are obtained in this way. In practice this means that a lot of tuples that would satisfy the constraint stated in line 24 are irrelevant.

Another way to improve the algorithm is to substitute the bisections in line 35 and 37 partly by the interval Newton method (while falling back to bisection, if the Newton method shows low progress). The Newton method is not only often faster than bisection, it also has the advantage of mostly only generating one successor interval instead of two, as bisection does. This keeps the cardinality of the sets L_k and R_k smaller, which in turn keeps the number of tuples to be considered in the loop of line 24 smaller.

A C++ implementation of the method sketched in Algorithm 1 can be found at `http://www2.informatik.uni-wuerzburg.de/mitarbeiter/tischler/`.

Two important implications of Theorem 3 are given in the following statements.

Theorem 4 *Let M be a complex square matrix of dimension n for $n \in \mathbb{N}, n > 0$ such that the symmetric real square matrix $S = M^H M$ (where M^H denotes the conjugate transpose of M) contains only rational numbers. Then it is decidable by applying Algorithm 1 to the characteristic polynomial of the matrix S whether the spectral norm of M is less than a given rational number d.*

Proof: The spectral norm $|M|_2$ of the matrix M is the maximum eigenvalue of the real symmetric positive semi-definite matrix $S = M^H M$. It is well known that $M^H M$ is diagonizable and has only real eigenvalues, i.e. the characteristic polynomial χ_S of S can be decomposed into linear factors and has only real roots. By construction χ_S has only rational coefficients. Thus according to Theorem 3 it is decidable using Algorithm 1, whether all roots of χ_S are smaller than d. □

Input : Real polynomial $p = \sum_{i=0}^{n} a_i x^i$ where $a_i \in \mathbb{Q}$ for $i = 0, \ldots, n$ and all roots of p are real, rational number d.
Output: **Yes** if and only if all roots of p are less than d.

if *p is constant* **then**
  if $p(d) = 0$ **then**
    Output$\leftarrow$**No**
  else
    Output$\leftarrow$**Yes**
else if $p(d) = 0$ **then**
  Output$\leftarrow$**No**
else
  Output$\leftarrow$**Undefined**
  $b \leftarrow$ positive root bound of p
  $L \leftarrow \{[-b, d]\}$
  $R \leftarrow \{[d, b]\}$
  while *Output=**Undefined*** **do**
    $L' \leftarrow \emptyset$
    $R' \leftarrow \emptyset$
    foreach $\mathcal{L} \in L$ **do if** $0 \in p(\mathcal{L})$ **then** $L' \leftarrow L' \cup \{\mathcal{L}\}$
    foreach $\mathcal{R} \in R$ **do if** $0 \in p(\mathcal{R})$ **then** $R' \leftarrow R' \cup \{\mathcal{R}\}$
    if $R' = \emptyset$ **then** Output$\leftarrow$ **Yes**
    else
      $L \leftarrow L'$
      $R \leftarrow R'$
      $g \leftarrow$ **No** // polynomial generated?
      // Assume that $L = \{\mathcal{L}_1, \ldots, \mathcal{L}_m\}$
      foreach $(n_1, \ldots, n_m) \in \mathbb{N}^m$ *such that* $\sum_{i=1}^{m} n_i = n$ **do**
        $p' = \sum_{i=0}^{n} A'_i x^i \leftarrow \prod_{i=1}^{m} (x - \mathcal{L}_i)^{n_i}$
        $g' \leftarrow$ **Yes**
        foreach $i = 0, 1, \ldots n$ **do**
          if $\{a_i\} \cap A'_i = \emptyset$ **then** $g' \leftarrow$ **No**
        if *g'=**Yes*** **then** $g \leftarrow$ **Yes**
      if *g=**No*** **then** Output$\leftarrow$ **No**
      else
        $L' \leftarrow \emptyset$
        $R' \leftarrow \emptyset$
        foreach $\mathcal{L} = [\underline{\mathcal{L}}, \overline{\mathcal{L}}] \in L$ **do**
          $L' \leftarrow L' \cup \left\{ \left[\underline{\mathcal{L}}, \frac{\overline{\mathcal{L}}+\underline{\mathcal{L}}}{2} \right] \right\} \cup \left\{ \left[\frac{\overline{\mathcal{L}}+\underline{\mathcal{L}}}{2}, \overline{\mathcal{L}} \right] \right\}$
        foreach $\mathcal{R} = [\underline{\mathcal{R}}, \overline{\mathcal{R}}] \in R$ **do**
          $R' \leftarrow R' \cup \left\{ \left[\underline{\mathcal{R}}, \frac{\overline{\mathcal{R}}+\underline{\mathcal{R}}}{2} \right] \right\} \cup \left\{ \left[\frac{\overline{\mathcal{R}}+\underline{\mathcal{R}}}{2}, \overline{\mathcal{R}} \right] \right\}$
        $L \leftarrow L'$
        $R \leftarrow R'$
return *Output*

Algorithm 1: Decision algorithm

Corollary 1 *Let M be a complex square matrix of dimension n for $n \in \mathbb{N}, n > 0$ such that the symmetric real square matrix $S = M^H M$ contains only rational numbers. Then it is decidable by applying Algorithm 1 to the characteristic polynomial of the matrix S whether M is a contraction mapping.*

5 Decomposition of complex polynomials

Unlike the real case, there are several possible choices concerning the shape of complex intervals, e.g. rectangular or circular. In this paper we use rectangular complex intervals, as they are suitable for our application.

The extension of the subdivision algorithm from the real to the complex case is straight-forward. We can decompose each non-empty rectangular interval into four equally sized quadrants such that the union of the quadrants is exactly the original interval. This is similar to the quad-tree decomposition of pixel images, as it is performed in WFA based image compression (cf. [7]). The size of a quadratic complex interval can be described by a single real number, which significantly simplifies the subsequent construction and discussion of algorithms.

We will for sake of simplicity assume that all rectangular intervals that are split by bisection operations are quadratic, i.e. their size can be described by a single real number. We refer to this number as the width of the interval. The interval evaluation of an expression for a quadratic interval is not necessarily a quadratic interval. However, we only compute evaluations to test, whether they contain the number 0. We do not split the computed intervals using bisection operations.

The check, whether an input polynomial p in Algorithm 1 has a root at d, i.e. the evaluation of p at a single point d, is critical. In the case of complex polynomials, there is no similar operation, i.e. if we have a complex interval B such that all roots of some polynomial are contained in B, we in general cannot decompose B into two intervals that touch in one single point and partition B. We can however still check, whether a set of complex intervals generates a complex polynomial. Thus we can state the following theorem.

Theorem 5 *Let $p(x)$ denote a complex polynomial where the real and imaginary parts of all coefficients are rational numbers. Let $\mathcal{L}$ and $\mathcal{R}_i$ denote complex intervals for $i = 1, \ldots, m$ for some natural number m such that all roots of p are contained in the union of $\mathcal{L}$ and all $\mathcal{R}_i$ and the intersection of $\mathcal{L}$ with every $\mathcal{R}_i$ is empty. Then it is decidable by using a generalization of Algorithm 1, whether $\mathcal{L}$ contains a root of p.*

Proof: We use an algorithm similar to Algorithm 1. By precondition $\mathcal{L}$ is disjoint from each $\mathcal{R}_i$, hence we cannot encounter sequences of intervals that converge to a point in an intersection of $\mathcal{L}$ and any $\mathcal{R}_i$. Thus we refine the intervals $\mathcal{L}$ and each $\mathcal{R}_i$, until either all intervals obtained from $\mathcal{L}$ have an interval evaluation that does not contain zero, or the intervals obtained from the $\mathcal{R}_i$ are no longer able to generate the polynomial p. □

We can extend this to check, whether some free-standing complex interval contains at least k roots for some natural number k.

Theorem 6 *Let $p(x)$ denote a complex polynomial where the real and imaginary parts of all coefficients are rational numbers and let k a natural number. Let $\mathcal{L}$ and $\mathcal{R}_i$ denote complex intervals for $i = 1, \ldots, m$ and some natural number m such that all roots of*

p are contained in the union of $\mathcal{L}$ and all $\mathcal{R}_i$ and the intersection of $\mathcal{L}$ with each $\mathcal{R}_i$ is empty. Then it is decidable by using a generalization of Algorithm 1, whether the number of roots contained in the interval $\mathcal{L}$ is k.

Proof: We again compute refinements of the given intervals like we do in Algorithm 1. When we try to generate a polynomial from a set of n intervals, we enumerate n-tuples of natural numbers (see Algorithm 1, line 24). When we observe the history of a computation, some intervals in the tuple will be descendants of the interval $\mathcal{L}$ and some of the intervals $\mathcal{R}_i$. As $\mathcal{L}$ and the $\mathcal{R}_i$ are disjoint, we will at some finite precision reach a point, where no interval descendant of some $\mathcal{R}_i$ is able to substitute any descendant of $\mathcal{L}$ and vice versa, when we produce a polynomial with interval coefficients and check whether the coefficients of p are contained in the corresponding intervals. Thus at some finite precision we will observe that for each tuple that generates p, the sum of the elements corresponding to descendants of the interval $\mathcal{L}$ is identical and equals some number κ. If this holds for one precision, then it holds for any greater precision. The number of roots in the interval $\mathcal{L}$ is k, if and only if $k = \kappa$. □

Finally, we can use this to formulate a polynomial decomposition result. The method employed is a simple extension of the one given in the proof of Theorem 6.

Theorem 7 *Let $p(x)$ denote a complex polynomial of degree n, where the real and imaginary parts of all coefficients are rational numbers and let ϵ denote a rational number. Then we can compute a natural number u and pairs of complex intervals and positive natural numbers (I_i, k_i) for $i = 1, \ldots, u$ such that*

- $\sum_{i=1}^{u} k_i = n$,
- $I_i \cap I_j = \emptyset$ *for* $(i, j) \in \{1, \ldots, u\}^2, i \neq j$,
- *the width of I_i is at most ϵ for $i = 1, \ldots, u$ and*
- *I_i contains exactly k_i (not necessarily distinct) roots of p*

using an extension of the method given in the proof of Theorem 6.

Note that using this method we are only only able to compute polynomial root approximations of arbitrary precision. The method does in general not provide the information, how many *distinct* roots of the polynomial are contained in each computed interval.

References

[1] G. Alefeld and J. Herzberger. Über Simultanverfahren Zur Bestimmung Reeller Polynomwurzeln. *Zeitschrift für Angewandte Mathematik und Mechanik*, 54:413–420, 1974.

[2] G. Alefeld and J. Herzberger. *Introduction to Interval Computations.* Academic Press, 1983.

[3] D. W. Arthur. The use of interval arithmetic to bound the zeros of real polynomials. *Journal of the Institute of Mathematics and its Applications*, 10:231–237, 1972.

[4] M. F. Barnsley. *Fractals everywhere.* Morgan Kaufmann, 2nd edition, 2000.

[5] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry.* Springer, 2003.

[6] H. Cohen. *A course in computational algebraic number theory*. Springer, 1993.

[7] K. Culik II and J. Kari. Inference algorithms for WFA and image compression. In Y. Fisher, editor, *Fractal Image Compression: Theory and Application*. Springer, 1995.

[8] J. R. Johnson. Real algebraic number computation using interval arithmetic. In P. S. Wang, editor, *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, pages 195–205, Berkeley, CA, 1992. ACM Press.

[9] U. Kulisch. *Computer Arithmetic and Validity*. de Gruyter, 2008.

[10] G. W. Walster. The extended real interval system. `http://www.mscs.mu.edu/~globsol/Papers/extended_intervals.ps`, 1998.