Available online at www.sciencedirect.com

Electronic Notes in Theoretical Computer Science 120 (2005) 145–157

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

# A Fast Algorithm for Julia Sets of Hyperbolic Rational Functions

Robert Rettinger [1]

*Department of Computer Science*
*FernUniversitaet Hagen*
*Hagen, Germany*

**Abstract**

Although numerous computer programs have been written to compute sets of points which claim to approximate Julia sets, no reliable high precision pictures of non-trivial Julia sets are currently known. Usually, no error estimates are added and even those algorithms which work reliable in theory, become unreliable in practice due to rounding errors and the use of fixed length floating point numbers.

In this paper we prove the existence of polynomial time algorithms to approximate the Julia sets of given hyperbolic rational functions. We will give a strict computable error estimation w.r.t. the Hausdorff metric on the complex sphere. This extends a result on polynomials $z \mapsto z^2 + c$, where $|c| < 1/4$, in [10] and an earlier result in [12] on the recursiveness of the Julia sets of hyperbolic polynomials.

The algorithm given in this paper computes Julia sets locally in time $O(k \cdot M(k))$ (where $M(k)$ denotes the time needed to multiply two $k$-bit numbers). Roughly speaking, the local time complexity is the number of Turing machine steps to decide a set of disks of spherical diameter $2^{-k}$ so that the union of these disks has Hausdorff distance at most $2^{-k+2}$. This allows to give reliable pictures of Julia sets to arbitrary precision.

*Keywords:* Julia Sets, Computational Complexity.

## 1 Introduction

Julia sets provide some of the most striking illustrations of how an apparently simple process can lead to highly intricate sets. Since they are a seemingly unexhaustable source of fantastic shapes and images (e.g. [9]), numerous

[1] Email: robert.rettinger@fernuni-hagen.de

1571-0661 © 2005 Elsevier B.V. Open access under CC BY-NC-ND license.
doi:10.1016/j.entcs.2004.06.041

computer programs have been written for generating pictures of Julia sets. Algorithms for computing approximations of Julia sets can be found e.g. in [11] and [8]. It has turned out that increasing precision (zooming) requires rapidly increasing computation time. Since usually the programs are realized on computers operating on floating point numbers of fixed length instead of real numbers, high precision pictures reflect effects of the rounding procedure rather than the intended Julia set. Therefore, presumably nobody has ever seen the micro-micro-micro structure of non-trivial Julia sets.

In this paper we give algorithms to compute the Julia sets of hyperbolic rational functions locally in time $O(k \cdot M(k))$ (where $M(k)$ denotes the time needed to multiply two $k$-bit numbers) on Turing machines, i.e. with a detailed analysis of rounding errors. Roughly speaking, the local time complexity is the number of Turing machine steps to give a (zoomed) set of pixels with Hausdorff distance at most $2^{-k+2}$. This gives the first polynomial bound on the complexity of such a general and rich class of Julia sets and extends a result of Rettinger and Weihrauch in [10], where a similar algorithm is given for polynomials $z \mapsto z^2 + c$ for $|c| < 1/4$. Furthermore, in [12] it is shown that the Julia sets of hyperbolic polynomials are recursive. However, the methods used in [12] cannot be applied to estimate computational complexity. In [6] it is shown that for fractal subsets of the Euclidean space computational complexity and Hausdorff dimension are independent.

In the next section we will give some basic general facts about Julia sets. In Section 3 we mainly discuss extensions to the local repelling property of $J$. This result allows us to use similar ideas to [10] even in the general case of rational functions. In Section 4 we then give and discuss the definition of local complexity on $\mathbb{C}_\infty$. Finally, in Section 5 we present our algorithms for Julia sets of hyperbolic rational functions. The algorithm for rational functions of degree at least 2 will be divided into a general preliminary phase, discussed in Subsection 5.1, and a main phase, discussed in Subsection 5.2.

## 2 Julia Sets

In this section we will give some basic notions and notations on Julia sets. Many of the results presented in this section are from the book [2] where many more details on Julia sets of rational functions can be found. Other valuable sources are [3], [5], [1], [8] and [7].

The set of rational, real and complex numbers are denoted by $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$, respectively. We consider polynomials $p(z) = a_n \cdot z^n + ... + a_0 \in \mathbb{C}[z]$ of one complex variable, where $a_n \neq 0$ is the leading coefficient and $n$ is the degree of $p$, and rational functions $r(z) = p(z)/q(z)$, where $p$ and $q$ are polynomials.

The degree of $r$ is the maximum of the degrees of $p$ and $q$, where we assume that $p$ and $q$ don't have any non-constant factors in common.

As usually we denote the $n$-th iterate and the derivative of a function $f$ by $f^n$ and $f'$, respectively. A complex number $z$ is called a fixed point of a function $f$ iff $f(z) = z$. $z$ is called periodic point of $f$ iff $z$ is a fixed point of some iterate of $f$. A fixed point $z$ is called repelling, attracting or indifferent iff $|f'(z)| > 1$, $|f'(z)| < 1$ or $|f'(z)| = 1$, respectively. Similarly, periodic points are classified (where $f$ have to be replaced by a sufficient iterate). We call $z$ a critical point of $f$ iff $f'(z) = 0$.

**Definition 2.1** The Julia set $J(r)$ of a rational function $r$ is the closure of all repelling periodic points of $r$.

There are many more equivalent characterizations of Julia sets, e.g. a point $z$ is in the Julia set of a rational function $r$ iff the sequence $\{r^n\}$ is not normal (or equivalently not equicontinuous) on some neighborhood of $z$. Another characterization is given below.

**Lemma 2.2** *Let $r$ be a rational function of degree at least 2. Then*

(1) *$J(r)$ is an uncountable compact set containing no isolated points.*

(2) *$r(J(r)) = r^{-1}(J(r)) = J(r) = J(r^n)$ for all integers $n > 0$.*

(3) *For any $z \in J(r), J(r) = cls(\bigcup_{k=1}^{\infty} r^{-k}(\{z\}))$ (where $cls(A)$ denotes the closure of the set $A$).*

(4) *For any neighborhood $U$ of $z \in J(r)$, and any compact set $K \subseteq \mathbb{C}$ there exist $z', z''$ and an integer $n > 0$ so that $r^n(U) \supset K \setminus \{z', z''\}$.*

Finally, we call a rational function $r$ hyperbolic iff the closure of the forward orbit $O_c(r) = \bigcup_{r'(z)=0} \bigcup_n r^n(z)$ of the critical points of $r$ is disjoint to $J(r)$. An immediate consequence of Lemma 2.2, (4) above is the following property of hyperbolic rational functions (see [2]). This will be the decisive ingredient for Lemma 3.2 in the next section and thus for our algorithms in Section 5.

**Lemma 2.3** *Let $r$ be a hyperbolic rational function and $c > 0$ be given. Then there exists a neighborhood $U$ of the Julia set $J(r)$ of $r$ and an integer $n > 0$ so that for all $m > 0$ and $z \in U$, $|(r^m)'(z)| > c$.*

Whereas polynomials can be treated adequately on the complex plane, an appropriate discussion of rational functions is only possible on the complex sphere $\mathbb{C}_\infty$ (see Section 3 below). There, all the above notions can be defined for the point $\infty$ in the usual way. We omit the details.

## 3 The Complex Sphere

Let $S$ denote the 2-sphere in $\mathbb{R}^3$, i.e. $S = \{(x, y, z) \in \mathbb{R}^3 : |x|^2+|y|^2+|z|^2 = 1\}$. Furthermore fix a point $\infty = (0, 0, 1)$. Then we can identify $S\setminus\{\infty\}$ with $\mathbb{C}$ by the stereographic projection $\phi^* : \mathbb{C} \to S$, $\phi^*(z) = \frac{1}{1+|z|^2}(2\Re(z), 2\Im(z), |z|^2 - 1)$, where $\Re(z)$ and $\Im(z)$ denote the real and imaginary part of $z$, respectively. Let $\sigma$ be the spherical metric on $\mathbb{C}$, i.e. $\sigma(z, z') = \inf\{\int_\gamma \frac{2}{1+|t|^2}|dt| :$ $\gamma$ is a smooth curve from $z$ to $z'\}$. This metric induces the usual metric on $S$ (viewed as manifold): $\sigma(z, z')$ is the length of the shortest path from $z$ to $z'$, where all points of the path have to be in $S$ (we will identify the complex numbers $z$ and $z'$ with $\phi^*(z)$ and $\phi^*(z')$ without further mentioning). We will denote $S$ together with the spherical metric by $\mathbb{C}_\infty$. As usually we use the notation $\sigma(z, A)$ for subsets $A \subset \mathbb{C}_\infty$ to denote the distance of a point $z$ to $A$, i.e. $\sigma(z, A) = \inf_{z'\in A} \sigma(z, z')$. Similarly we define $\sigma(A, A')$ for $A, A' \subset \mathbb{C}$ by $\sigma(A, A') = \max\{\sup_{z\in A} \sigma(z, A'), \sup_{z\in A'} \sigma(z, A)\}$.

It is obvious that every rotation of the sphere induces a $\sigma$ isometry of $\mathbb{C}_\infty$. These rotations are exactly those Möbius transformations $M(a, c)$, $M(a, c)(z) = \frac{az-c}{\bar{c}z+\bar{a}}$, with $|a|^2+|c|^2 = 1$, which maps $\frac{c}{a}$ to 0. Furthermore for each $z \in \mathbb{C}\setminus\{0\}$ there exists a unique representation $z = \frac{c}{a}$, where $|a|^2 + |c|^2 = 1$ and $a$ is a non-negative real number. Let $M_z := M(a, c)$, where $\frac{c}{a}$ is this unique representation, $M_\infty(z) := z^{-1}$ and $M_0$ be the identity $id$ on $\mathbb{C}_\infty$.

We will give the main result of this section in Lemma 3.2 below. We prepare this result by some remarks. The following (rough) relation of the spherical and Euclidean metric will be used frequently below.

**Lemma 3.1** *Let $z, z' \in \mathbb{C}$ and $\gamma$ be the geodesic from $z$ to $z'$, $\infty \notin \gamma$. Then*

$$|z - z'| \leq (1 + \beta_\gamma^2)^2 \cdot \sigma(z, z') \text{ and } \sigma(z, z') \leq 2|z - z'|$$

*where $\beta_\gamma = \sup_{u\in\gamma} |u|$.*

Notice, that in the above lemma $\beta_\gamma$ exists because the image of $\gamma$ (which we again denote by $\gamma$) is compact. We will switch several times between spherical and Eucledean metric in the proof of Lemma 3.2 below. To simplify notations we will use $\beta_K^- := \sup_{u\in K} |u|$, $\beta_K := \sup_{u\in K} |r(u)|$, $\beta'_K := \sup_{u\in K} |r'(u)|$ and $\beta''_K := \sup_{u\in U} |r''(u)|$ for compact sets $K \subset \mathbb{C}$.

As the Euclidean and the spherical metric coincide in 0, we can transfer local properties from $\mathbb{C}$ to $\mathbb{C}_\infty$ by using suitable Möbius transformations. For example, in the proof of Lemma 3.2 we will need derivatives on $\mathbb{C}_\infty$: Given a rational function $r$, let $r'_\infty(z)$ be determined by $(M_{r(z)} \circ r \circ M_z^{-1})'(0)$. A simple calculation gives $r'_\infty(z) = \frac{1+|z|^2}{1+|r(z)|^2} r'(z)$ for $z \in \mathbb{C}$. To bound the variation of $r'_\infty(z)$ for different points $z$, we would also like to define something like a second derivative $r''_\infty$. Unfortunately, however $r'_\infty$ is not analytic. To circumvent this problem, we define for $\delta > 0$, $r''_\delta(z) := (r_\delta)'_\infty$, where $r_\delta = (1+\delta^2)r'(z)$ (the meaning of this will become clear in the proof below). One easily verifies $r''_\delta(z) = r''(z)(1+\delta^2)(1+|z|^2)/(1+|r(z)|^2)$.

**Lemma 3.2** *There exists a polynomial $P$ so that for all rational functions $r$ and all complex numbers $z$, $z'$ with $|z-z'| < 1/2$ and $|r(z)-r(z')| < 1/2$ we have*

$$(|r'_\infty(z)| - P(\beta^-_\gamma, \beta_\gamma, \beta'_\gamma, \beta''_\gamma)\sigma(z,z'))\sigma(z,z') \le \sigma(r(z), r(z'))$$

*and*

$$\sigma(r(z), r(z')) \le (|r'_\infty(z)| + P(\beta^-_\gamma, \beta_\gamma, \beta'_\gamma, \beta''_\gamma)\sigma(z,z'))\sigma(z,z').$$

*where $\gamma$ denotes the geodesic from $z$ to $z'$, i.e. $\sigma(z,z') = \int_\gamma 2/(1+|z|^2)|dz|$, and $\infty \notin \gamma \cup r(\gamma)$.*

**Proof.** Let $r$, $\gamma$ and $z, z'$ be given according to the lemma and let $K \supset \gamma$ be a sufficiently small compact neighborhood of $\gamma$ not containing $\infty$. We will achieve the above result in several steps. First we show, that we can replace $r$ by the rational function $q = M_{r(z)} \circ r \circ M_z^{-1}$ which simplifies the calculation of the spherical distance between $z, z'$ and $r(z), r(z')$.

**Claim 3.3** *Let $q = M_{r(z)} \circ r \circ M_z^{-1}$ and $z_\infty = M_z(z')$. Then*

*a) $q'_\infty(0) = q'(0) = r'_\infty(z)$, $\quad q(0) = 0$,*

*b) $\sigma(q(0), q(z_\infty)) = \sigma(r(z), r(z')$, $\quad \sigma(0, z_\infty) = \sigma(z,z')$ and*

*c) for all $v \in M_z(K)$ we have $|q(v)| \le 1$, $\quad |q'(v)| \le 16\beta'_K(1+|r(z)|)^2$ and*

*$|q''(v)| \le 2^9(1+|r(z)|)^4(\beta''_K + \beta'_K + (\beta'_K)^2)$.*

We will use $q$ instead of $r$ in the sequel. To simplify matters we reuse $\gamma$ to be the geodesic from 0 to $z_\infty$. Next we rewrite $\sigma(r(z), r(z'))$ by $\sigma(r(z), r(z')) = \sigma(q(0), q(z_\infty)) = \int_0^{q(z_\infty)} \frac{2}{1+|u|^2}|du| = \int_0^{z_\infty} \frac{2|q'(u)|}{1+|q(u)|^2}|du| = \int_0^{z_\infty} \frac{2|q'_\infty(u)|}{1+|u|^2}|du|$ where we use $\int_a^b f(u)du$ for the integral on the straight line from $a$ to $b$, for short. Next we bound $|q'_\infty(0) - q'_\infty(u)|$ by a constant and the spherical distance of 0 and $u$. To state this formally, we need another technical result.

**Claim 3.4** *There exists an $\varepsilon > 0$ so that for all $t, t'$, so that the straight line from $\gamma(t)$ to $\gamma(t')$ belongs to $K$, we have for $\Delta = |\gamma(t') - \gamma(t)|$*

$$\left|\frac{1+|\gamma(t)|^2}{1+|q(\gamma(t)|^2} - \frac{1+|\gamma(t')|^2}{1+|q(\gamma(t')|^2}\right| \le (2\beta'_K(1+(\beta^-_K)^2) + 2\beta^-_K)\Delta + \varepsilon\Delta^2.$$

Now let $u$ be a point on $\gamma$ and denote the part of $\gamma$ from 0 to $u$ again by $\gamma$. Obviously, $\gamma$ is a geodesic. Furthermore let $\Delta > 0$ be given and $t_0, ..., t_n$ be a partition of the domain of $\gamma$ so that $|\gamma(t_i) - \gamma(t_{i+1})| \le \Delta$ and the straight line from $\gamma(t_i)$ to $\gamma(t_{i+1})$ belongs to $K$. With $\Delta_i := |\gamma(t_{i+1}) - \gamma(t_i)|$ and $\delta_i := \beta'_K(2\beta'_K(1 + (\beta^-_K)^2) + 2\beta^-_K)\Delta_i + \varepsilon\Delta_i^2$ we have

$$\begin{aligned} |q'_\infty(u) - q'_\infty(0)| &\leq \textstyle\sum_{i=0}^{n-1} |q'_\infty(t_{i+1}) - q'_\infty(t_i)| \\ &\leq \textstyle\sum_{i=0}^{n-1} \int_{\gamma(t_i)}^{\gamma(t_{i+1})} |q''(v)(\frac{1+|\gamma(t_i)|^2}{1+|q(\gamma(t_i))|^2})||dv| + \delta_i \end{aligned}$$

Taking the limit $\Delta \to 0$ leads to

$$|q'_\infty(u) - q'_\infty(0)| \leq (|q''_\infty| + \beta'_K(2\beta'_K(1 + (\beta^-_K)^2) + 2\beta^-_K))|u|$$

where $|q''_\infty| \leq \beta''_K \cdot (1+(\beta^-_K)^2)$ is $|q''_\infty| = \sup_{v\in K} |q''_\infty(v)|$. Putting things together we can finally conclude

$$\sigma(r(z), r(z')) = \int\limits_0^{z_\infty} \frac{2|q'_\infty(u)|}{1 + |u|^2}|du| = q'_\infty(0)\sigma(0, z_\infty) + \varepsilon$$

where

$$|\varepsilon| \leq \textstyle\int_0^{z_\infty} 2|\frac{q'_\infty(u) - q'_\infty(0)}{1+|u|^2}||du| \leq (|q''_\infty| + \beta'_K(2\beta'_K(1 + (\beta^-_K)^2) + 2\beta^-_K))|u|\sigma(0, u).$$

By Lemma 3.1 we can bound $|u|$ by $\sigma(0, u)$ and a suitable factor. As this holds for arbitrary $K$ the proof is completed.

□

## 4 Local Complexity on $\mathbb{C}_\infty$

The local complexity of subsets of $\mathbb{C}$ was introduced in [10]. Given a Turing machine $M$, let $L(M)$ denote the set accepted by $M$ and let $U_M(k)$ be the union of all intervals $I_{k,i,j} = [2^{-k} \cdot i; 2^{-k} \cdot (i + 1)] \times [2^{-k} \cdot j; 2^{-k} \cdot (j + 1)]$ with "$0^k, i, j$" $\in L(M)$, where $i, j$ are given by their binary representation. Then we say, that $M$ decides a closed set $A_M$ iff for each $k$, $U_M(k)$ covers $A_M$ and additionally has Hausdorff distance at most $2^{-k+1}$ to $A_M$. Notice that $A_M$ is uniquely determined by $M$. Identifying the subsets $I_{k,i,j}$ (for given $k$) with a pixel $(i, j)$ (say on a monitor), this definition is probably quite accurate to investigate the problem of drawing pictures of subsets up to arbitrary precisions. Finally, we say that a set $A \subseteq \mathbb{C}$ has local complexity $O(t)$ iff there exists a $O(t)$ time bounded Turing machine $M$ with $A_M = A$.

This rises the question why we don't investigate in a "global complexity" of a set, e.g. the complexity of giving all the pixels in $U_M(k)$. The answer is simply that such a global complexity is not suitable for most of the problems: For many sets we need an exponentially number (in $k$) of pixels of size $2^{-k}$ to cover the set. This would lead to exponentially time bounded algorithms, hiding the finer complexity structures. Furthermore we cannot display an unbounded number of pixels on a monitor of fixed resolution. Thus, what we usually do, is to zoom into the picture, and the complexity of this zooming is quite accurately captured by the notion of local complexity given above.

This notion does neither depend on the exact shape of the pixels nor on the Hausdorff distance $2^{-k+1}$. The latter can be chosen to be $(\sqrt{2^{-k}} + \varepsilon) \cdot 2^{-k}$ for arbitrary $\varepsilon > 0$. More generally, for each $k$, let $C_k = (V_{i,k})_i$ be a covering of $\mathbb{C}$ of diameter at most $d \cdot 2^{-k}$. Then, for given $k$ and $\varepsilon > 0$, any set $A \subset \mathbb{C}$ can be covered by a union $V_k = \bigcup_{j \in I} V_{j,k}$, where $I \subseteq \mathbb{N}$, so that the Hausdorff distance of $A$ and $V_k$ is at most $(d+\varepsilon)2^{-k}$. Thus, replacing the intervals $I_{k,i,j}$ by $V_{i,k}$ gives a more general notion of local complexity. If we replace herein the Euclidean metric by the spherical metric, we get a suitable definition for the local complexity of subsets of $\mathbb{C}_\infty$. To simplify matters we fix $V_{i,k}$ to be a circle with suitable center. Furthermore we fix $\varepsilon := 1$.

**Definition 4.1** For given $k, i, j \in \mathbb{N}$ let

$$c_{k,i,j} = \begin{cases} (\frac{i}{2} \cdot 2^{-k-2}, j \cdot 2^{-k-2}, \sqrt{1 - (|\frac{i}{2}|^2 - |j|^2)2^{-2k-4}}) & \text{if } i \text{ is even} \\ (\frac{i-1}{2} \cdot 2^{-k-2}, j \cdot 2^{-k-2}, -\sqrt{1 - (|\frac{i-1}{2}|^2 - |j|^2)2^{-2k-4}}) & \text{if } i \text{ is odd} \end{cases}$$

and $C_{k,i,j} = \{z \in \mathbb{C}_\infty : c_{k,i,j} \in \mathbb{C}_\infty \wedge \sigma(z, c_{k,i,j}) \leq 2^{-k}\}$. A Turing machine decides a set $A \subseteq \mathbb{C}_\infty$ iff $A_M := \bigcup_{``0^k,i,j" \in L(M)} C_{k,i,j} \supseteq A$ and $\sigma(A, A_M) \leq 2^{-k+2}$ for all $k \in \mathbb{N}$, where $i, j$ are presented to $M$ by their binary representation. A subset $A \subseteq \mathbb{C}$ has local complexity $O(t)$ iff there exists a $O(t)$ time bounded Turing machine $M$ which decides $A$.

For compact subsets of $\mathbb{C}$ the local complexity (as subset of $\mathbb{C}_\infty$) and the complexity defined in [10] coincide. Thus again we could draw parts of the approximations according to the discussion given above, i.e. according to the Euclidean metric, as long as we are far enough away from $\infty$. However, a probably better way of presenting such pictures is by means of local spherical projections. Similar to the complex plane we can represent any set in $\mathbb{C}_\infty$ by approximations demanded by the above definition:

**Lemma 4.2** *For every set $A \subseteq \mathbb{C}_\infty$ and every $k \in \mathbb{N}$ there is a set $I \subseteq \mathbb{N} \times \mathbb{N}$ so that $A_I := \bigcup_{(i,j) \in I} C_{i,j,k} \supset A$ and $\sigma(A, A_I) < 2^{-k+2}$.*

## 5 A Fast Algorithm for Rational Functions

In the sequel we use a Turing machine model with extra (oracle) tapes and states, so that entering these states with a string of the form $0^n \sharp 0^i \sharp 0^j \sharp 0^s$ results in an answer $x \sharp y \sharp 0^d$ where $x$ and $y$ are the binary representation (say of length $\log_2\lfloor x \rfloor + n + 2$) of an approximation of the imaginary and real part of the $i$-th linear factor (up to $2^{-n}$) of the nominator ($s = 0$) or denominator ($s = 1$) polynomial. Furthermore $d$ gives the degree of this polynomial. We don't assume a certain order on the linear factors so that this representation is equivalent to the representation by the coefficients of the polynomials. Although the representation is one among many others, it seems to be natural for many problems in practice.

The algorithms presented in the sequel assume, that the rational function $r = p/q$ represented by the (input) polynomials $p$ and $q$ are hyperbolic and furthermore $p$ and $q$ have no factors in common. Notice, that neither of these assumptions can be decided for the given representation of the input.

Our algorithms work in two phases. In the preliminary phase a first approximation to the Julia set is calculated. This phase outputs three open sets $U_0 \subset U_1 \subset U_2$, each represented by a finite sequence of open balls with rational center and radius, a bijective Möbius transformation $\tau$ and $\delta > 0$, $n > 0$ so that for the $n$-th iteration $s = (\tau \circ r \circ \tau^{-1})^n$ we have

$(i)$ $\tau(z_r) \notin K := \{z : |z| \leq \delta\}$ for a critical point $z_r$ of $r$,

$(ii)$ $J' := J(s)(= \tau(J(r))) \subseteq U_0 \subset U_2 \subset K$,

$(iii)$ $\sigma(U_2, J') \leq \varepsilon^{-1}$, $\sigma(z, \mathbb{C}_\infty \setminus U_2) \geq \frac{1}{3}\varepsilon^{-1}$ and $\sigma(z', \mathbb{C}_\infty \setminus U_1) \geq \frac{1}{3}\varepsilon^{-1}$ for all $z \in U_1, z' \in U_0$,

$(iv)$ $|(s)'_\infty(z)| \geq 2$ for all $z \in U_2$,

where $\varepsilon = P(\beta_K^-, \beta_K, \beta'_K, \beta''_K)$ ($P$ the polynomial from Lemma 3.2). This phase does not depend on the requested precision or pixel.

In the main phase we will then approximate the Julia set up to the requested precision $2^{-k}$ for a pixel $C_{k,i,j}$ in the sense of Definition 4.1. This phase needs time $O(k \cdot M(k))$, where $M(k)$ is the time needed to multiply two $k$ bit numbers. Using the Schönhage-Strassen algorithm we get a (local) complexity of $O(n^2 \log n \log\log n)$ for the Julia set of a hyperbolic rational function. Notice however, that, despite the fact, that our algorithms are uniform in the sense, that they compute the Julia sets for any given hyperbolic rational function, the given time bound is not uniformly, i.e. the constant factor hidden in the big-O depends on the given function. Furthermore we strongly conjecture, that a common polynomial or even exponential time bound on the complexity of Julia sets of hyperbolic functions of a certain degree $\geq 2$ does not exist.

### *5.1 Preliminary Computations*

In this section we give an informal description on how to compute $U_0$ to $U_2$, $\tau$ and $\delta$. By the representation of the rational function $r = p/q$ it is clear, that the value of $r$ and the modulus of continuity (to arbitrary precision) can be determined at any complex number $z = x + iy$, $x$, $y$ rational and $q(z) \neq 0$.

Furthermore for given points $z$ and $z'$ in $\mathbb{C}_\infty$, it is possible to compute the spherical distance to arbitrary precision, for example by using suitable Möbius transformations.

It is quite easy to give fast algorithms for rational functions of degree 1: In this case there exists bijective Möbius transformations $\phi$ so that $\phi \circ r \circ \phi^{-1}$ is of the form $z \mapsto z+\beta$ or $z \mapsto kz$. In the former case $J(r) = \emptyset$ and in the latter $J(r)$ depends on $k$: if $|k| < 1$ then $J(r) = \{\infty\}$, if $|k| > 1$ then $J(r) = \{0\}$, if $|k| = 1$ we have $J(r) = \emptyset$. For more details see [2]. We will thus assume, that the rational functions given to our algorithms are of degree at least 2.

Now to the preliminary computations which we need in Subsection 5.2. In a first step we compute an $2^{-k}$ (say $k = 3$) approximation to $\tau$. If $r$ is a polynomial we simply take $\tau = id$. Otherwise we can find an approximation $z_c$ to a critical point of $r$. Let $\tau := M_\infty \circ M_{z_c}$ and $s := \tau \circ r \circ \tau^{-1}$. Notice that this is a first approximation to $\tau$. If we later recognize, that $\tau$ is not sufficient, we increase $k$ and compute a better approximation to $z_c$ and then again set $\tau = M_\infty \circ M_{z_c}$ and repeat the following steps for this new $\tau$.

Next we search an approximation of a repelling periodic point $z_0$ of $s$. If we fail after a fixed number of steps (say e.g. $100 \cdot k$), restart with a better approximation for $z_c$ and $\tau$. By Lemma 2.2 and Definition 2.1 we will finally find such a point and this point belongs to $J(r)$.

Then we compute one after the other the sets $s^{-1}(z_0)$, $s^{-1}(s^{-1}(z_0))$, ..., i.e. the preimages of $z_0$, the preimages of the preimages of $z_0$ and so on. At the same time we try to find a set $A$ of open balls $B(z)$ of points $z$ in these preimages and an index $n$ so that a) $z_0 \in \bigcup_{U(z)\in A} U(z)$, b) for each $U(z) \in A$ we have $(s^n)^{-1}(U(z)) \subseteq \bigcup_{U(z)\in A} U(z)$, c) the radius of $U(z)$ is at most $\varepsilon^{-1}$ and d) $|(s^n(z'))'_\infty| > 2$ for all $z' \in \bigcup_{U(z)\in A} U(z)$. If we find such an $n$ and a set $A$, we are done. If we fail to find such a set $A$ after a fixed number of steps (say again $100 \cdot k$) we will restart with a better approximation for $z_c$ and $\tau$.

Once such a set $A$ is found we can take $U_2 = \bigcup_{U(z)\in A} U(z)$ and repeat this procedure to get $U_1$ and finally $U_0$. The existence of such an $A$ follows by the following result.

**Lemma 5.1** *For each $\delta > 0$ there exists an $n > 0$ and a finite set $A$ of balls $U(z) = \{z' : \sigma(z, z') < r_z\}$ of radius $r_z \leq \delta$, so that*

$$J(r) \subset \bigcup_{U(z)\in A} ((\tau \circ r \circ \tau^{-1})^n)^{-1}(U(z)) \subset \bigcup_{U(z)\in A} U(z).$$

A few comments on this procedure are in order: As we can compute the sets mentioned above only up to finite precision, the properties a) to d) are meant to be guaranteed by the algorithm. Lemma 5.1 gives enough space to do this by appropriate approximations. Another problem are the roots of $q$.

If we are near such roots, nearly nothing can be said about the modulus of continuity. This problem is resolved by choosing a suitable $\tau$.

### 5.2 *Approximation up to Arbitrary Precision*

We will now assume that the properties (i) to (iv) of the introductory discussion of this section are fulfilled, i.e. $n, \delta, \tau$ and $U_0, U_1, U_2$ are chosen properly.

The problem to decide, on input $0^k, i, j$, wether the pixel $C_{k,i,j}$ should belong to a covering $A$ of $J(r)$ of Hausdorff distance at most $2^{-k+2}$ is equivalent to the decision, wether the pixel $\{z : \sigma(z, \tau(c_{k,i,j})) < 2^{-k}\}$ should belong to a covering $A'$ of $\tau(J(r))$ of Hausdorff distance at most $2^{-k+2}$. Thus, to simplify matters, we replace $r$ by $(\tau \circ r \circ \tau^{-1})^n$ presented by the preliminary phase, and $i, j$ by the corresponding values, where we reuse the same notations $r$, $i, j$ again. (Actually we have to replace the single pixel by a finite number of pixels of higher precision, as the new center is probably not a center of a pixel $C_{k,i',j'}$. We omit the purely technical details.)

Furthermore we divide the question, wether the pixel $C_{k,i,j}$ is near to $J(r)$ into tests $T_{k,i',j'}(2^{-k}/t) \in \{0, 1\}$ where $t > 1$ and

$$T_{k,i',j'}(2^{-k}/t) = \begin{cases} 1 & \text{if } \sigma(c_{k+k',i',j'}, J(r)) < 2^{-k}/t \\ 0 & \text{if } \sigma(c_{k+k',i',j'}, J(r)) > 2^{-k} \end{cases}$$

for a suitable constant $k'$.

Obviously, we can then cover and thus decide the pixel $C_{k,i,j}$ by at most $(4 \cdot t)^2$ such tests $T_{k,i',j'}(2^{-k}/t)$. To simplify notation, we use $c_{k,i,j}$ for the center of these tests, although they are actually of the form $C_{k',i,j}$ for some $k' > k$. Notice, that different to the algorithm in [10] we have to cover the whole surface of a pixel by such tests, as the Julia set might no longer be connected.

To finally apply Lemma 3.2 and give an algorithm for the tests $T_{k,i,j}(2^{-k}/t)$, we have to consider the errors introduced by rounding (intermediate) results. Let therefore $\delta = \beta_K^-$ be the constant of the introductory discussion of this section, i.e. $U_2 \subset K = \{z : |z| < \beta_K^-\}$ and $l_0 := \lceil \log_2 \beta_K^- + 1 \rceil$. Furthermore, for $x \in K \cap \mathbb{R}$, let $[x]_l$ denote an $l_0 + l$ bit approximation of $x$ so that $|x - [x]_l| \leq 2^{-l+1}$ and the fractional part of $[x]_l$ has $l$ bits. We will denote the set of all such $[x]_l$ with $x \in K$ by $K_l$, for short. Similarly, for $z \in \mathbb{C}$ we use $[z]_l$ to denote roundings of the real and imaginary part of $z$. The corresponding set is thus $K_l + iK_l$. Obviously, we have $|z - [z]_l| \leq 2^{-l+2}$ and thus $\sigma(z, [z]_l) \leq 2^{-l+3}$. Furthermore, we define the product $[\prod_{s=1}^n]_l x_s$ for $x_1$, ..., $x_n$ in $K_l$ by $[\prod_{s=1}^0]_l x_s = 1$ and $[\prod_{s=1}^{t+1}]_l x_s = [x_{t+1} \cdot [\prod_{s=1}^t]_l x_s]_l$. The following lemma, which can be proven by a simple induction, gives a rough upper bound on the error made by multiplying in this way (in a special case).

**Lemma 5.2** *Let $N, l \in \mathbb{N}$, $\delta > 0$ and $x_1$, ..., $x_N$ in $K_l$ with $|x_i| \leq \delta$ (for all $i$) be given. Then $|[\prod_{s=1}^{N}]_l x_s - \prod_{s=1}^{N} x_s| \leq (2\delta + 1)^N \cdot 2^{-l}$.*

Let $K$ and $\varepsilon = P(\beta_K^-, \beta_K, \beta_K', \beta_K'')$ be given as in the introductory discussion of this section. Furthermore let $k, m > 0$ and $t > 1$ be given. Then we can use Lemma 3.2 to prove the following version of this lemma, respecting rounding errors.

**Lemma 5.3** *Let $l = k + 6 + \lceil \log_2 t \rceil + \lceil \log(\beta_K'(1 + (\beta_K^-)^2) + 1/m) \rceil - \lfloor \varepsilon \rfloor$ and $z \in K_l + iK_l$ be given so that $\sigma(z, J(r)) \leq 2/(3m\varepsilon)$. Then either*

$$\sigma(z, J(r)) \leq (\beta_K'(1 + \beta_K^2) + 1/m)^{-1} 2^{-k-2}/t \text{ and } \sigma([r(z)]_l, J(r)) \leq 2^{-k}/t$$

*or*
$$\sigma(z, J(r)) > (\beta_K'(1 + \beta_K^2) + 1/m)^{-1} 2^{-k-2}/t, \text{ and}$$
$$(|r_\infty'(z)| - (1/m))\sigma(z, J(r)) \leq \sigma([r(z)]_l, J(r)) \leq (|r_\infty'(z)| + (1/m))\sigma(z, J(r))$$

Now we are prepared to give the main algorithm. The following algorithm performs for given $k' \geq 2$ a test $T_{k,i,j}(2^{-k}/t)$ with the help of a constant number of pixels of a $2/(3 \cdot 2^{k'}\varepsilon)$ approximation to $J(r)$. To simplify notation let $N(k) = \lceil k/\log_2(2/3) \rceil + 1$ and $L(k, k') = \max\{k + 6 + \lceil \log_2 t \rceil + \lceil \log(\beta_K'(1 + (\beta_K^-)^2) + 2^{-k'}) \rceil - \lfloor \varepsilon \rfloor, N(k) \lceil \log_2(2\beta_K'(1 + (\beta_K^-)^2) + 1) \rceil\}$, thus $L(k, k') \leq c \cdot (k + k')$ for some constant $c > 0$ (not depending on $k$).

```
ALGORITHM Reduce(k')
    INPUT: $0^k, i, j$
    OUTPUT: 0 or $z_n$ (see Lemma 5.4 below)
BEGIN
    $z_0 := c_{k,i,j}$, $l := L(k, k')$, $n := 0$, $m := 1$;
    WHILE ($z_n \in U_1$ and $m \leq 1 + 2^{k+1}/(3 \cdot 2^{k'}\varepsilon)$) DO
      $m := [[|r_\infty'(z_n)|]_l \cdot m]_l$; $z_{n+1} := [r(z_n)]_l$, $n := n + 1$;
    END WHILE
    IF $z_n \notin U_1$ THEN RETURN 0
    ELSE RETURN $z_n$
END
```

The correctness of the above algorithm is given by the following lemma.

**Lemma 5.4** *The above algorithm stops on all inputs $c_{k,i,j}$. If the algorithm returns 0 then $\sigma(c_{k,i,j}, J(r)) > 2^{-k}/t$ and otherwise the algorithm returns $z_n$ with $n \leq N(k)$ and*

$$T_{k,i,j}(2^{-k}/t) = \begin{cases} 1 & \text{if } \sigma(z_n, J(r)) \leq 2/(3 \cdot 2^{k'}\varepsilon) \\ 0 & \text{if } \sigma(z_n, J(r)) \geq 2(\beta_K'(1 + (\beta_K^-)^2) + 2)/(3 \cdot 2^{k'}\varepsilon) \end{cases}$$

*is a test for $c_{k,i,j}$ in the sense discussed above, where*

$$t = 3 \cdot (1 + 2^{-k'})^{N(k)} / (1 - 2^{-k'})^{N(k)}$$

**Proof.** By Lemma 5.2 we have for $n \le N(k)$:

$$|\prod_{s=0}^{n}(|r'_\infty(z_s)| \cdot (1 - 2^{-k'}) - [[\prod_{s=0}^{n}]_l(|r'_\infty(z_s)| \cdot (1 - 2^{-k'})]_l| \le 1$$

and thus $[[\prod_{s=0}^{N(k)}]_l(|r'_\infty(z_s)| \cdot (1 - 2^{-k'})]_l| \ge (\frac{3}{2})^{N(k)} - 1 \ge 2^k$. That means, the algorithm must have stopped before reaching $z_{N(k)}$. For $z_0$ with $\sigma(z_0, J(r)) \ge 2^{-k}$ we get by Lemma 5.3 for all computed $z_n$

$$\begin{aligned}\sigma(z_n, J(r)) &\ge (\textstyle\prod_{s=0}^{n-1}(|r'_\infty(z_s)| \cdot (1 - 2^{-k'}))\sigma(z_0, J(r)) \\ &\ge ([[(\textstyle\prod_{s=0}^{n-1}]_l(|r'_\infty(z_s)| \cdot (1 - 2^{-k'}))]_l - 1)\sigma(z_0, J(r))\end{aligned}$$

For the final $z_n$ returned by the algorithm we have therefore either $z_n \notin U_1$ or $\sigma(z_n, J(r)) \ge 2/(3 \cdot 2^{k'}\varepsilon)$. Analogously, we get the upper bound.

□

Using $(1 + 1/n)^n \le 3$ and $(1 - 1/n)^n \ge 1/3$ for $n > 1$ we get for $k' = \lceil \log_2 N(k) \rceil$ immediately.

**Corollary 1** *If Reduce($\lceil \log_2 N(k) \rceil$) returns $z_n$ then*

$$T_{k,i,j}(2^{-k}/c) = \begin{cases} 1 & \textit{if } \sigma(z, J(r)) \le 2/(3N(k)\varepsilon) \\ 0 & \textit{if } \sigma(z, J(r)) \ge 2(\beta'_K(1 + (\beta^-_K)^2) + 2)/(3N(k)\varepsilon) \end{cases}$$

*is a test for $c_{k,i,j}$ in the sense discussed above, where $c$ is a suitable constant not depending on $k$. Otherwise we have $\sigma(c_{k,i,j}, J(r)) > 2^{-k}/c$.*

Notice that for $k' = 2$ there exists a constant $c$ (not depending on $k$) so that $t$ in the above lemma can be bounded by $2^{cN(k)}$. Furthermore the above algorithm needs in each step a finite number of multiplications, divisions and additions of $l + c' \le c(k + k') + c'$-bit numbers (the additive constant stems from the fact, that we have to compute intermediate results up to a higher precision to get the demanded precision of the result in the considered step). Thus for $k' \le k$ the above algorithm needs at most $c \cdot k \cdot M(k) + c$ steps to present a result, where $c$ does not depend on $k$.

Now the main result can be deduced easily: First use algorithm Reduce(2) and the set $U_1$ to get an exponentially time bounded algorithm for the local complexity of $J(r)$ (by the discussion above we need $(4 \cdot t)^2$ tests of the form $T_{k,i,j}$, i.e. $O(2^{2cN(k)}kM(k))$ steps, to decide for a given pixel $C_{k,i,j}$ wether it should be accepted or not). The idea of a faster algorithm is similar to the one in [10]. There it is shown that, for given $t$, $c' > c$ and any algorithm which decides $J(r)$ in time $O(t)$, one can decide the question wether $\sigma(z, J(r)) \le c2^{-k})$ or $\sigma(z, J(r)) \ge c'2^{-k}$ in time $O(t(c''k + c''))$ for a suitable constant $c''$ not depending on $k$. Thus by Corollary 1 we can use first the algorithm Reduce($\lceil \log_2 N(k) \rceil$) and decide the reduced problem afterwards by the exponentially time bounded algorithm Reduce(2). This leads to a polynomially time bounded algorithm, as the length of the reduced input is $O(\log_2 N(k)) = O(\log_2 k)$, compared to the original input length $O(k)$. Using Reduce($\lceil \log_2 N(k) \rceil$) and this polynomially time bounded algorithm then gives the following result.

**Theorem 5.5** *The Julia sets of hyperbolic rational functions have a local complexity of $O(k \cdot M(k))$, where $M(k)$ denotes the time needed to multiply two $k$-bit numbers.*

## References

[1] M. F. Barnsley, *Fractals Everywhere*, Academic Press, Boston, 2nd edition, 1993.

[2] A. F. Beardon, *Iteration of Rational Functions*, Springer, 1991.

[3] R. L. Devaney, *An introduction to chaotic dynamical systems*, Addison-Wesley, Redwood City, 2nd edition, 1989.

[4] A. Douady, *Does a Julia set depend continuously on the polynomial?*, Proc. Symposia in Applied Math., Complex and Dynamical Systems: The Mathematics Behind the Mandelbrot Set and Julia Sets, 49:91-138, 1994.

[5] K. Falconer, *Fractal Geometry*, Mathematical Foundations and Applications, John Wiley & Sons, Chichester, New York, 1990.

[6] H. Kamo, K. Kawamura and I. Takeuti, *Computational Complexity of fractal sets*, Real Analysis Exchange, 26(2):773-793, 2000/01.

[7] J. Milnor, *Dynamics in One Complex Variable*, Vieweg, Braunschweig, 2nd edition, 2000.

[8] H.-O. Peitgen, H. Jürgens and D. Saupe, *Fractals for the classroom, Part Two, Complex Systems and the Mandelbrot set*, Springer, 1992.

[9] H.-O. Peitgen and P. H. Richter, *The beauty of fractals. Images of complex dynamical systems*, Springer, Berlin, 1986.

[10] R. Rettinger and K. Weihrauch, *The Computational Complexity of Some Julia Sets*, Proceedings of the 35th Annual ACM Symposium on Theory of Computing, 177-185, San Diego, 2003.

[11] D. Saupe, *Efficient computation of Julia sets and their fractal dimension*, Physica, 28D:358-370, 1987.

[12] N. Zhong, *Recursively enumerable subsets of $\mathbb{R}^q$ in two computing models: Blum-Shub-Smale machine and Turin machine*,Theoretical Computer Science, 197:79-94,1998.