Available online at www.sciencedirect.com
SCIENCE DIRECT
ELSEVIER
Electronic Notes in Theoretical Computer Science 120 (2005) 17–30
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

Hyperbolic Julia Sets are Poly-Time Computable

Mark Braverman[1,2]

Dept. of Computer Science
University of Toronto
Toronto, ON, Canada

Abstract

In this paper we prove that hyperbolic Julia sets are locally computable in polynomial time. Namely, for each complex hyperbolic polynomial $p(z)$, there is a Turing machine $M_{p(z)}$ that can "draw" the set with the precision 2^{-n}, such that it takes time polynomial in n to decide whether to draw each pixel. In formal terms, it takes time polynomial in n to decide for a point x whether $d(x, J_{p(z)}) < 2^{-n}$ (in which case we draw a pixel with center x), or $d(x, J_{p(z)}) > 2{\cdot}2^{-n}$ (in which case we don't draw this pixel). In the case $2^{-n} \leq d(x, J_{p(x)}) \leq 2 \cdot 2^{-n}$ either answer will be acceptable. This definition of complexity for sets is equivalent to the definition introduced in Weihrauch's book [16] and used by Rettinger and Weihrauch in [13].
Although the hyperbolic Julia sets were shown to be recursive, complexity bounds were proven only for a restricted case in [13]. Our paper is a significant generalization of [13], in which polynomial time computability was shown for a special kind of hyperbolic polynomials, namely, polynomials of the form $p(z) = z^2 + c$ with $|c| < 1/4$.
We show that the machine drawing the Julia set can be made independent of the hyperbolic polynomial p, and provide some evidence suggesting that one cannot expect a much better computability result for Julia sets.
We also introduce an alternative real set computability definition due to Ko, and show an interesting connection between this definition and the main definition.

Keywords: computable analysis, Julia sets, computational complexity, complex dynamics.

[1] Research is partially supported by the Natural Sciences and Engineering Research Council of Canada.

[2] Email: mbraverm@cs.toronto.edu

doi:10.1016/j.entcs.2004.06.031

1 Introduction

Nowadays, computers are being increasingly applied to represent mathematical objects. Computer-generated images are being extensively used in the analysis and simulations of real-life processes and their mathematical models. Our goal is to investigate a formal framework which allows us to define the computational complexity of real sets, measuring the complexity of drawing the set on a computer. Within this framework, we obtain a new result on the computability of Julia sets.

We mainly use the definition of real set complexity introduced by Weihrauch in [16] and used in [13] as the measure of complexity of some Julia sets (see also [2]).

In sections 2 and 3 we present two different definitions of computability of real sets that have been proposed, and show that they are equivalent if and only if P=NP, a result of independent interest. Theorem 3.3 can be used to prove computability of many sets for which a direct proof of computability would be hard.

Julia sets are some of the best known illustrations of a highly complicated chaotic system generated by a very simple mathematical process. These sets have been deeply studied in the framework of complex dynamics during the last century. Julia sets are not only an intriguing mathematical object, but also a major source of amazing images. Many computer programs, some of which are freely available on the web, have been written to generate these images. Algorithms for computing Julia sets have been presented and discussed in [11] and [14], for example.

It appears, however, that none of the algorithms or their implementations cope well with zooming in. With the computer using fixed-precision numbers, rounding errors significantly affect the computation when we try to zoom in. These programs also seem to work poorly near some "pathological" polynomials, for example, with $p(z) = z^2 + 1/4 + \varepsilon$, $0 < \varepsilon \ll 1$. We will return to this example in section 8.

We give the first polynomial bound on the complexity of an arbitrary hyperbolic Julia set. The class of hyperbolic polynomials is very rich. For example, in the case $p(z) = z^2 + c$, $p(z)$ is hyperbolic for all c's outside the Mandelbrot set. It is conjectured that it is also hyperbolic for all c's in the interior of the Mandelbrot set (but not on the boundary), see [9] for more information. The algorithm is outlined in sections 6 and 7. The details of the construction are mathematically involved, and many of them had to be omitted due to space constraints.

The algorithm that we present is not uniform in $p(z)$. That is, the Turing