COMPUTATIONAL COMPLEXITY OF FRACTALS

KER-I KO

Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794
E-mail: keriko@cs.sunysb.edu

This paper surveys recent work on the computability and computational complexity of two-dimensional regions and fractal curves, in the context of Turing machine-based computable analysis and discrete computational complexity theory.

1. Introduction

Fractals are interesting objects that occur in many different areas of research (see, e.g., Mendelbrot [11], Peitgen et al. [12], Barnsley [1] and Falconer [5]). Simple iterative algorithms can produce complicated fractals in the limit. These fractals, though easy to generate, often have fine structures that make their basic properties hard to compute. For instance, Mendelbrot [11], in his study of the coastline problem, pointed out that it may be easy to generate a Koch-like two-dimensional fractal curve but hard to measure its length. In order to understand the precise relations between the computational complexity of a fractal curve itself and that of its other properties, such as its length and the area of its interior, we need to develop a formal theory of computational complexity of fractals.

In this paper, we present a short survey of recent work toward this goal. We apply the formal theory of computable analysis and computational complexity of real functions, which use Turing machines as the basic computational model, to the study of fractals. In particular, we discuss how to extend these theories to define computable and feasibly computable two-dimensional regions. Based on this computational model, we then consider some basic issues about fractals, including the following questions:

(1) If a two-dimensional curve Γ is computable (or polynomial-time computable), does it follow that its length is also computable (or, respectively, polynomial-time computable)?

(2) If a two-dimensional Jordan curve (i.e., a simple closed curve) Γ is computable (or polynomial-time computable), does it follow that the area of its interior is also computable (or, respectively, polynomial-time computable)?
(3) If a two-dimensional curve Γ is computable (or polynomial-time computable), does it follow that its Hausdorff dimension is also computable (or, respectively, polynomial-time computable)?
(4) If a two-dimensional curve Γ is computable (or polynomial-time computable) and its Hausdorff dimension is also computable (or polynomial-time computable), does it follow that its Hausdorff measure is also computable (or, respectively, polynomial-time computable)?
(5) If a function f is computable (or polynomial-time computable), does it follow that its (generalized) Julia set $J(f)$ is also computable (or, respectively, polynomial-time computable)?

For most of these questions, we present a negative answer. That is, through the construction of various fractal curves, we show that there exist feasibly computable curves (or functions) whose related properties are not even computable.

In the following, we assume basic knowledge of computable analysis and complexity theory of real functions. For a complete treatment of these subjects, see Pour-El and Richards [13], Weihrauch [17], Ko [6] and [8]. For the basic notions of complexity classes discussed in this paper, see Du and Ko [4].

2. Computability of a Two-Dimensional Region

First, we present a brief review of the notion of computable real functions. We use Cauchy sequences of dyadic rational numbers to represent real numbers. Let $\mathbb{D}$ denote the set of dyadic rationals; i.e.,

$$\mathbb{D} = \Big\{ m + \sum_{i=1}^{n} d_i 2^{-i} \ \Big|\ m \in \mathbb{Z}, d_1, d_2, \ldots, d_n \in \{0,1\} \Big\}.$$

Each dyadic rational d has an infinite number of binary representations. For instance, $d = 3/8$ may be represented by

$$0.011\underbrace{00\cdots0}_{k}$$

with an arbitrary number k of trailing zeroes. A binary representation s of a dyadic rational d is said to have n *bits (in the fractional part)* if s has exactly n bits to the right of the binary point. We let D denote the set of binary representations of dyadic rationals, and let D_n be the set of all binary representations $s \in D$ which have exactly n bits. When there is no confusion, we use $s \in D$ to denote both a binary representation in D and the dyadic rational d it represents.

We say a function $\phi : \mathbb{N} \to \mathbb{D}$ *binary converges* to a real number x, or *represents a real number* x, if (i) for all $n \geq 0$, $\phi(n)$ has a binary representation of at most n bits, and (ii) for all $n \geq 0$, $|x - \phi(n)| \leq 2^{-n}$. For each function ϕ that binary converges to x, there is a corresponding set representation of x, namely, the set

$$L_\phi = \{s \in D \mid (\exists n)\,[s \text{ has } n \text{ bits and } s < \phi(n)]\}.$$

(Note that L_ϕ is defined as a set of binary representations of dyadic rationals, instead of a set of dyadic rationals. It is possible that two different $s, t \in D$ represents the same dyadic rational d, but one is in L_ϕ and the other is not.) We call such a set a *left cut* of x.

For any $x \in \mathbb{R}$, there is a unique function $\phi_x : \mathbb{N} \to \mathbb{D}$ that binary converges to x and satisfies the condition $x - 2^{-n} \leq \phi_x(n) < x$ for all $n \geq 0$. We call this function ϕ_x the *standard Cauchy function* for x. The left cut L_{ϕ_x} that corresponds to the standard Cauchy function ϕ_x for x is called the *standard left cut* of x, and we write L_x for L_{ϕ_x}. It is easy to see that $L_x = \{s \in D \mid s < x\}$. (Note that the set L_x may be treated as a set of dyadic rationals. The membership of a dyadic rational d in L_x is independent of its binary representations.)

To compute a real-valued function $f : \mathbb{R} \to \mathbb{R}$, we use oracle Turing machines as the computational model. A function $f : \mathbb{R} \to \mathbb{R}$ is said to be *computable* if there is an oracle Turing machine M, that uses a function $\phi : \mathbb{N} \to \mathbb{D}$ that binary converges to a real number x as an oracle, such that on input $n \in \mathbb{N}$, $M^\phi(n)$ outputs a string $d \in D_n$ such that $|d - f(x)| \leq 2^{-n}$.

This notion of computable real functions can be extended to functions from $\mathbb{R}$ to $\mathbb{R}^2$ (and functions from $\mathbb{R}^2$ to $\mathbb{R}^2$) in a natural way. More precisely, a function $f : \mathbb{R} \to \mathbb{R}^2$ is *computable* if there is an oracle Turing machine M that, on oracle ϕ which binary converges to $x \in \mathbb{R}$ and an input $n \in \mathbb{N}$, outputs two strings $d_1, d_2 \in D_n$ such that both $|d_1 - y_1|$ and $|d_2 - y_2|$ are bounded by 2^{-n}, where $f(x) = \langle y_1, y_2 \rangle$. To compute a function f from $\mathbb{R}^2$ to $\mathbb{R}^2$, we modify the above oracle Turing machine M to use two oracle functions $\phi_1, \phi_2 : \mathbb{N} \to \mathbb{D}$ which binary converge to two real numbers x_1 and x_2.

We now extend this concept of computability to two-dimensional subsets

$S \subseteq \mathbb{R}^2$. For any set $S \subseteq \mathbb{R}^2$, let χ_S denote its characteristic function; i.e., $\chi_S(\langle x, y\rangle) = 1$ if $\langle x, y\rangle \in S$ and $\chi_S(\langle x, y\rangle) = 0$ if $\langle x, y\rangle \notin S$. Intuitively, a set $S \subseteq \mathbb{R}^2$ is *computable* if χ_S is a computable function, or, equivalently, if there is an oracle Turing machine M that uses two functions that represent a point $\langle x, y\rangle$ in $\mathbb{R}^2$ as oracles and decides whether $\langle x, y\rangle \in S$. However, we notice that the function χ_S is discontinuous at the boundary of S, and hence, from a well-known result in computable analysis, it cannot be decided completely by the Turing machine M (unless S is trivial). Instead, we must allow M to make mistakes. There are several ways to relax the requirement on machine M. Here, we present two formal definitions.

The first one is *recursive approximability*, which is an extension of the notion of recursive approximability of sets of real numbers in Ko [6]. Informally, a set $S \subseteq \mathbb{R}^2$ is recursively approximable if there is an oracle Turing machine M that takes two functions representing a point $\langle x, y\rangle \in \mathbb{R}^2$ as the oracles and an integer n as the input, and decides whether $\langle x, y\rangle$ is in S in such a way that the errors of M occur in a set $E_n(M) \subseteq \mathbb{R}^2$ of Lebesgue measure less than 2^{-n}.

We write $\mu(A)$ to denote the two-dimensional Lebesgue measure of a set $A \subseteq \mathbb{R}^2$, and $\mu^*(A)$ the outer measure of set A.

Definition 2.1. A set $S \subseteq \mathbb{R}^2$ is *recursively approximable* if there exists an oracle Turing machine M such that for any oracles (ϕ, ψ) representing a point $\langle x, y\rangle \in \mathbb{R}^2$ (i.e., ϕ and ψ binary converge to x and y, respectively), and for any input n, M outputs either 0 or 1 such that the following set $E_n(M)$ has measure $\mu^*(E_n(M)) \le 2^{-n}$:

$$E_n(M) = \{\langle x, y\rangle \in \mathbb{R}^2 \mid \text{there exists } (\phi, \psi) \text{ representing } \langle x, y\rangle \text{ such that } M^{\phi,\psi}(n) \neq \chi_S(\langle x, y\rangle)\}.$$

The notion of recursively approximable sets is equivalent to the notion of *recursively measurable* sets of Šanin [15].

The second notion of computability of a two-dimensional subset of $\mathbb{R}^2$ is *recursive recognizability*. Informally, a set $S \subseteq \mathbb{R}^2$ is recursively recognizable if there exists a Turing machine that recognizes whether two given oracle functions represent a point $\langle x, y\rangle$ in S correctly for all points that have a distance greater than 2^{-n} from the boundary of S, where n is the input.

Definition 2.2. A set $S \subseteq \mathbb{R}^2$ is *recursively recognizable* if there exists an oracle Turing machine M that works as in Definition 2.1 such that $E_n(M) \subseteq \{\langle x, y\rangle \in \mathbb{R}^2 \mid$ the distance between $\langle x, y\rangle$ and the boundary of S is $\le 2^{-n}\}$.

Both of the above two computability notions of two-dimensional regions allow *two-sided errors* to occur in the computation; that is, errors may occur at some point $\langle x_1, y_1\rangle \in S$ and also at some point $\langle x_2, y_2\rangle \notin S$. For certain types of regions, we might want to consider the notion of computability with only one-sided errors. For instance, if a set S has Lebesgue measure equal to zero, then it is trivially recursively approximable, since an oracle Turing machine that always outputs 0 has error measure $\mu(E_n(M)) = 0$. Such an oracle Turing machine also recursively recognizes S since all the errors occur at points in S and hence at the boundary of S. To more correctly reflect the notion of computability on such sets, we would like the oracle Turing machine to have errors only when the oracles represent a point $\langle x, y\rangle$ that is not in S. We call a recursively approximable (or recursively recognizable) set $S \subseteq \mathbb{R}^2$ *strongly recursively approximable* (or, respectively, *strongly recursively recognizable*) if the underlying oracle Turing machine M satisfies the additional condition that $E_n(M) \cap S = \emptyset$.

The particular type of subsets of the two-dimensional plane we are interested in here is the class of bounded simple regions, i.e., bounded, connected subsets of $\mathbb{R}^2$ which contain no holes. For simple regions whose boundaries are simple closed curves (i.e., Jordan curves), we have a third natural notion of computability: the computability of its boundary as a function from $[0, 1]$ to $\mathbb{R}^2$. That is, we may consider a region S *computable* if there is a computable real function $f : [0, 1] \to \mathbb{R}^2$ whose range is exactly the boundary Γ_S of region S.

The relations between these notions of computability on two-dimensional regions are not simple. We first list some positive results.

Theorem 2.1. (Chou and Ko [3]) *(a) If the boundary of a region S is a computable Jordan curve, then S is recursively recognizable.*

(b) If the boundary of a region S is a computable Jordan curve of a finite length, then S is recursively approximable.

(c) If S is recursively approximable, then S is recursively recognizable.

The proofs of the following negative results about these computability notions are based on the fractal construction of Section 4.

Theorem 2.2. *(a) There exists a computable Jordan curve Γ whose interior S is not recursively approximable.*

(b) There exists a recursively recognizable set S that is not recursively approximable.

For the more precise relation between a computable Jordan curve and recursive approximability of its interior, see Theorem 4.1(b).

Finally, we remark that most relations between strongly recursively approximable sets and strongly recursively recognizable sets are negative ones. We summarize these results below. Since the proofs are relatively simple and do not involve the construction of fractal curves, we omit them here.

Theorem 2.3. (Chou and Ko [3]) *(a) There exists a recursively approximable set that is not strongly recursively approximable.*

(b) There exists a recursively recognizable set that is not strongly recursively recognizable.

(c) The class of strongly recursively approximable sets and the class of strongly recursively recognizable sets are incomparable.

3. Computational Complexity of Two-Dimensional Regions

Since the notion of computable real functions is defined by oracle Turing machines, it is natural to extend this notion to polynomial-time computable real functions. Namely, we say a function $f : \mathbb{R} \to \mathbb{R}$ is *polynomial-time computable* if f is computable by an oracle Turing machine M such that for all oracles ϕ and all inputs n, $M^{\phi}(n)$ halts in time $p(n)$ for some polynomial function p.

Polynomial-time approximable and polynomial-time recognizable sets $S \subseteq \mathbb{R}^2$ can also be defined based on the time complexity of oracle Turing machines.

Definition 3.1. (a) A set $S \subseteq \mathbb{R}^2$ is *polynomial-time approximable* if it is recursively approximable by an oracle Turing machine M that, on input n, runs in time $p(n)$ for some polynomial p, regardless of what the oracles are.

(b) A set $S \subseteq \mathbb{R}^2$ is *polynomial-time recognizable* if it is recursively recognizable by an oracle Turing machine M that, on input n, runs in time $p(n)$ for some polynomial p, regardless of what the oracles are.

The notions of strongly polynomial-time approximable sets and strongly polynomial-time recognizable sets can be defined in a similar way. Strongly polynomial-time recognizable sets are interesting since they can be used to characterize the sets of zeroes of polynomial-time computable functions from $[0,1]^2$ to $\mathbb{R}^2$. (A function $f : [0,1]^2 \to \mathbb{R}^2$ is *polynomial-time computable* if it is computable by a two-oracle Turing machine M such that for all oracles ϕ, ψ and all inputs n, $M^{\phi,\psi}(n)$ halts in time $p(n)$ for some polynomial function p.) Rettinger and Weihrauch [14] also used the notion of *local time complexity*, which is closely related to the notion of strongly polynomial-time recognizability, to study the computational complexity of Julia sets.

Regarding the relations between these notions of polynomial-time computability of two-dimensional regions, we note that results in Theorem 2.1 do not extend to their polynomial-time counterpart immediately. They depend on the unknown relations between discrete complexity classes. To avoid the definitions of some nonstandard complexity classes, we only present the relations in terms of well-known complexity classes. For more details, see Chou and Ko [3]. The following complexity classes of sets of strings are well known:

- P: the class of sets computable in polynomial time by deterministic Turing machines;
- FP: the class of functions computable in polynomial time by deterministic Turing machines;
- BPP: the class of sets computable in polynomial time by probabilistic Turing machines, with bounded errors;
- NP: the class of sets computable in polynomial time by nondeterministic Turing machines;
- $\#P$: the class of functions that count the number of accepting computation paths of a polynomial-time nondeterministic Turing machine;
- UP: the class of sets computable in polynomial time by nondeterministic Turing machines that have at most one accepting computation path.

It is known that $P \subseteq UP \subseteq NP$, $P \subseteq BPP$, and $FP \subseteq \#P$. It is also known that $[FP = \#P] \Rightarrow [P = NP]$, and $[FP = \#P] \Rightarrow [P = BPP]$. It is not known whether any of the above equations holds.

Theorem 3.1. (Chou and Ko [3]) *In the following, (a) $\Rightarrow$ (b) $\Rightarrow$ (c).*

(a) $FP = \#P$.

(b) All polynomial-time approximable subsets of $[0,1]^2$ are polynomial-time recognizable.

(c) $BPP = P$.

Theorem 3.2. (Chou and Ko [3]) *In the following, $(a) \Rightarrow (b) \Rightarrow (d)$ and $(a) \Rightarrow (c) \Rightarrow (d)$.*

(a) $FP = \#P$.

(b) If Γ is a polynomial-time computable Jordan curve, then the interior S of Γ is polynomial-time recognizable.

(c) If Γ is a polynomial-time computable Jordan curve, and if Γ has a finite length, then the interior S of Γ is polynomial-time approximable.

(d) $UP = P$; i.e., one-way functions do not exist.

In other words, if $BPP \neq P$ and $UP \neq P$, then the relations of Theorem 2.1 do not hold for their polynomial-time counterparts.

To demonstrate the proof techniques for these results, we present in the following, a sketch of the proof of a weaker result of $(b) \Rightarrow (c)$ of Theorem 3.1.

Theorem 3.3. *If there exists a set $T \subseteq \{1\}^*$ (called a* tally set*) in the class $BPP - P$, then there is a nondegenerate rectangle $[a,b] \times [c,d]$ that is polynomial-time approximable but not polynomial-time recognizable.*

Sketch of Proof. The main observation is that there is a simple construction of a real number x from a given tally set $T \subseteq \{1\}^*$ such that x and T have the same time complexity, within a polynomial factor. Namely, define

$$x = \sum_{n=1}^{\infty} (\chi_T(1^n) + 1) \cdot 4^{-n}.$$

Then, it can be proved that x and T satisfy the following properties:

(1) The standard left cut L_x of x is polynomial-time computable relative to T (i.e., it can be computed in polynomial time by an oracle Turing machine that uses set T as an oracle).

(2) Set T is polynomial-time computable relative to x (i.e., T can be computed in polynomial time by an oracle Turing machine that uses a function ϕ that binary converges to x as an oracle).

Thus, for $T \in BPP - P$, we know that x is not polynomial-time computable but its standard left cut L_x is in BPP. Let A be the rectangle $[0,x] \times [0,1]$. We claim that A satisfies the required conditions.

First, we check that A is not polynomial-time recognizable. Assume, by way of contradiction, that A is polynomial-time recognizable by an oracle Turing machine M. Then, we can compute an approximation d to x within error 2^{-n} as follows: We simulate M to binary search for a dyadic rational $e \in \mathbb{D}$ which has a binary representation in D_{n+1} such that $M^{\langle e, 1/2 \rangle}(n+1) = 1$ and $M^{\langle e', 1/2 \rangle}(n+1) = 0$, where $e' = e + 2^{-(n+1)}$. Then, either M is correct on these two computations and hence $e \leq x \leq e'$, or M makes a mistake on at least one of these two computations and hence e or e' is within distance $2^{-(n+1)}$ of the right line segment of the rectangle. In either case, we know that $|x - e| \leq 2^{-n}$ and $|x - e'| \leq 2^{-n}$. One of the numbers e or e' has a binary representation in D_n and is the required approximation. This contradicts the fact that x is not polynomial-time computable.

Conversely, we check that A is polynomial-time approximable. Let M_1 be a *probabilistic* Turing machine that accepts the standard left cut L_x of x in polynomial time. It means that there is a two-input *deterministic* Turing machine $\tilde{M}_1$ and a polynomial function p such that, for all dyadic rationals $d \in D_n$,

(a) if $d < x$ then $\Pr_{|w|=p(n)}[\tilde{M}_1(d,w) = 1] > 1 - 2^{-n}$, and

(b) if $d > x$ then $\Pr_{|w|=p(n)}[\tilde{M}_1(d,w) = 1] < 2^{-n}$.

(In the above, we have used a standard method of *amplification of accepting probability* on probabilistic Turing machine M_1; see Theorem 8.19 of Du and Ko [4].)

Now,we design an oracle Turing machine M as follows: For any oracle functions ϕ, ψ that represent a point $\langle y, z\rangle$ in $[0,1]^2$, and for any input n, it first gets $d_y = \phi(n+3+p(n+3))$ and $d_z = \psi(n+3)$. Then, it decodes d_y to d and w where d consists of the first $n+3$ bits of d_y and w is the string of the last $p(n+3)$ bits of d_y (i.e., in the binary expansion form, $d_y = 0.dw$). M accepts if and only if $\tilde{M}_1(d,w) = 1$ and $0 \le d_z \le 1$. It is not hard to verify that M only makes mistake at a point $\langle y, z\rangle$ if (1) $|z-1| \le 2^{-(n+3)}$ or $|z-0| \le 2^{-(n+3)}$, (2) $|y-x| \le 2^{-(n+3)}$ or $|y-0| \le 2^{-(n+3)}$, (3) $d_y = 0.dw$ and $d < x$ but $\tilde{M}_1(d,w) = 0$ or (4) $d_y = 0.dw$ and $d > x$ but $\tilde{M}_1(d,w) = 1$. The outer measure of these error areas is, for each of (1) and (2), at most $2^{-(n+2)}$ and, for (3) and (4) together, at most $2^{-(n+2)}$. Altogether, we have $\mu^*(E_n(M)) \le 2^{-n}$. □

The negative results of Theorem 2.2 can be extended to the polynomial-time setting.

Theorem 3.4. *(a) There exists a polynomial-time computable Jordan curve Γ whose interior S is not recursively approximable.*

(b) There exists a polynomial-time recognizable set $S \subseteq \mathbb{R}^2$ that is not recursively approximable.

For strongly polynomial-time computability notions of two-dimensional regions, all negative results of Theorem 2.3 still hold.

Theorem 3.5. [Chou and Ko, 3] *(a) There exists a polynomial-time approximable set that is not strongly polynomial-time approximable.*

(b) There exists a polynomial-time recognizable set that is not strongly polynomial-time recognizable.

(c) The class of strongly polynomial-time approximable sets and the class of strongly polynomial-time recognizable sets are incomparable.

4. Fractal Curves and Their Interiors

In this section, we consider Questions (1) and (2) listed in Section 1. We first give a complete characterization of the computability of the interior of a Jordan curve in terms of the Lebesgue measure of the curve itself.

Recall that a set $S \subseteq \mathbb{R}^2$ is *r.e. open* if there exist recursive functions $\phi, \psi, \theta : \mathbb{N} \to \mathbb{D}$ such that $S = \bigcup_{n=1}^{\infty} R_n$, where R_n is the $\theta(n) \times \theta(n)$ open square centered at $\langle \phi(n), \psi(n) \rangle$. We say $T \subseteq \mathbb{R}^2$ is *r.e. closed* if $\mathbb{R}^2 - T$ is r.e. open. A real number x is called *left r.e.* if its standard left cut L_x is r.e.; it is called *right r.e.* if $-x$ is r.e.

The relations between the computability of a Jordan curve Γ and the computability of its interior S and its measure $\mu(S)$ can be summarized as follows:

Theorem 4.1. (Ko and Weihrauch [10]) *(a) If a Jordan curve Γ is computable, then its interior S and its exterior T are r.e. open sets. Furthermore, the measure $\mu(S)$ must be a left r.e. real number, and the two-dimensional Lebesgue measure $\mu(\Gamma)$ must be a right r.e. real number.*

(b) If a Jordan curve Γ is computable and the two-dimensional Lebesgue measure $\mu(\Gamma)$ of the curve itself is zero, then its interior S is recursively approximable.

(c) If a Jordan curve Γ is computable and the two-dimensional Lebesgue measure $\mu(\Gamma)$ of the curve itself is a recursive real number, then the measure $\mu(S)$ of its interior S must be a recursive real.

It is interesting to note that, in Theorem 4.1(b) and 4.1(c), the curve Γ may be a fractal (a fractal curve may have measure zero in the two-dimensional plane). That is, even if a curve Γ is a fractal, its interior is still recursively approximable—as long as the two-dimensional measure of Γ is zero.

On the other hand, if the curve Γ has a nonrecursive two-dimensional measure (and hence must be a fractal), then it may have a noncomputable interior.

Theorem 4.2. (Ko and Weihrauch [10]) *For any left r.e. real number $x > 0$, there is a polynomial-time computable Jordan curve Γ whose interior has Lebesgue measure equal to x.*

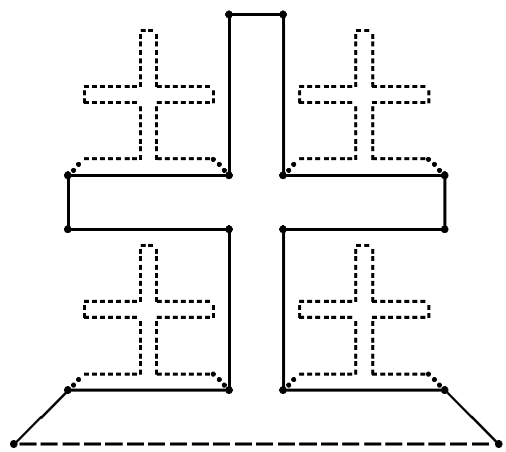

Figure 1. A basic curve of Γ_n. The dash line is a basic line segment of Γ_{n-1}, and the dot lines are the basic curves of Γ_{n+1} that replaces the basic line segments of Γ_n.

Proof Idea. The proof of this result is based on the construction of a *monster* curve. (A curve is called a monster curve if it is a simple curve and yet its two-dimensional measure is greater than zero [11].) It is a standard iterative construction that is used to define, for instance, the Koch curve. Namely, we start with a line segment Γ_0. Then we construct at each stage $n > 0$, a curve Γ_n, and let Γ be the limit of Γ_n. Each curve Γ_n has 4^n *basic line segments*, and Γ_n is constructed from Γ_{n-1} by replacing each basic line segment of Γ_{n-1} by a *basic curve* that contains four shorter basic line segments (see Figure 1). By choosing a suitable initial basic line segment Γ_0, we can make the limit curve Γ of Γ_n to have a positive measure q for any given dyadic rational $q > 0$. Correspondingly, the interior of Γ has a measure $r \in \mathbb{D}$.

Intuitively, if x is left r.e., then there is a Turing machine M that outputs an increasing sequence $d_1, d_2, \ldots$ of dyadic rationals that converges to x. If we simulate this machine for i moves, and output the largest d_j that M generates in i moves, this sequence can be generated by a *polynomial-time* Turing machine. Formally, if x is left r.e., then there exists a *polynomial-time computable* function $e : \mathbb{N} \to \mathbb{N}$ such that $e(i) \leq 2i$ and

$$x = \sum_{i=0}^{\infty} \operatorname{sgn}(e(i)) \cdot 2^{-e(i)},$$

where $\operatorname{sgn}(m) = 1$ if $m \geq 1$ and $\operatorname{sgn}(m) = 0$ otherwise.

To prove the theorem, we modify the above construction of Γ_{n+1} as follows: At stage n, we select $2^{2n-e(n)}$ basic line segments of Γ_n. On each basic line segment we selected, we replace the line segment with a suitable curve which does not include any basic line segment. On each remaining basic line segment, we replace the line segment with a basic curve of Γ_{n+1}.

In the original construction of the monster curve Γ, the limit of the basic curves of $\Gamma_n, \Gamma_{n+1}, \ldots$ has a positive measure $q/2^{2n}$ around each basic line segment of Γ_n. Thus, the interior of Γ loses measure $q/2^{2n}$ to its boundary around this basic line segment. Since Γ_n has 2^{2n} basic line segments, the interior of Γ totally loses measure q. Now, our modification reduces the number of basic line segments in Γ_n from 2^{2n} to $2^{2n-e(n)}$, so this modification gains measure $q \cdot 2^{-e(n)}$ for the interior. In the limit, we gain measure qx for the interior, where $x = \sum_{n\geq 1} \mathrm{sgn}(e(n)) \cdot 2^{-e(n)}$. Thus, the measure of the interior of the limit curve becomes $r + qx$. $\square$

Since there exist left r.e. real numbers that are not recursive, and since the Lebesgue measure of a recursively approximable region in $\mathbb{R}^2$ must be a recursive real [3], we have the following negative result regarding the first half of Question (2) of Section 1. This also proves Theorems 2.2 and 3.4.

Corollary 4.1. *There exists a polynomial-time computable Jordan curve whose interior has a nonrecursive two-dimensional Lebesgue measure.*

For the second half of Question (2), we can show a stronger result: For a polynomial-time computable Jordan curve, its interior does not have to be polynomial-time approximable, even if the Lebesgue measure of the curve itself is zero. The proof is based on a similar iterative construction of a curve Γ, with the modification that the Hausdorff dimension of the curve is actually equal to one (and hence the curve is not a fractal) but the length of the curve is still infinite. (We delay the formal definition of Hausdorff dimensions until Section 5.)

Theorem 4.3. (Ko and Weihrauch [10]) *For any recursive real number $x > 0$, there is a polynomial-time computable Jordan curve Γ whose two-dimensional Lebesgue measure is zero and whose Hausdorff dimension is equal to one, such that its interior has Lebesgue measure equal to x.*

Corollary 4.2. *(a) There exists a polynomial-time computable Jordan curve Γ whose two-dimensional Lebesgue measure is zero and whose Hausdorff dimension is equal to one, such that its interior has a measure that is not polynomial-time computable.*

(b) There exists a polynomial-time computable Jordan curve Γ whose two-dimensional Lebesgue measure is zero and whose Hausdorff dimension is equal to one, such that its interior S is not polynomial-time approximable.

Note that the length of the curve Γ in Theorem 4.3 and Corollary 4.2 is infinite. This property of curve Γ is critical. In fact, if a Jordan curve Γ is of a finite length, then the Lebesgue measure of its interior must be polynomial-time computable relative to a function in $\#P$.

Theorem 4.4. *The following are equivalent:*

(a) Each function $f : \{1\}^* \to \mathbb{N}$ that belongs to $\#P$ is actually in FP.

(b) For each polynomial-time approximable set $S \subseteq [0, 1]^2$, the measure $\mu(S)$ is polynomial-time computable.

(c) If a Jordan curve Γ is polynomial-time computable and has a finite length, then the measure $\mu(S)$ of its interior S is also polynomial-time computable.

Finally, we note that a simple modification of the construction of Theorem 4.2 yields a negative result for Question (1) of Section 1.

Theorem 4.5. (Ko [7]) *There exists a polynomial-time computable Jordan curve Γ that has finite length, but its length is a left r.e., nonrecursive real number.*

5. Hausdorff Dimensions and Hausdorff Measure

A fractal is usually defined as a set whose Hausdorff dimension is strictly higher than its topological dimension (see Mandelbrot [11]). First let us review the notion of Hausdorff dimensions.

Let A be a subset of $\mathbb{R}^n$. A *δ-covering* of A is a countable collection $\{U_i\}_{i=1}^{\infty}$ of sets in $\mathbb{R}^n$ such that $A \subseteq \bigcup_{i=1}^{\infty} U_i$ and $|U_i| \le \delta$, for $i \ge 1$, where $|U_i| = sup\{|x-y| \mid x, y \in U_i\}$. For any real number $s > 0$, the *s-dimensional Hausdorff measure* of A is defined as

$$\mathcal{H}^s(A) = \liminf_{\delta \to 0} \Big\{ \sum_{i=1}^{\infty} |U_i|^s \mid \{U_i\}_{i=1}^{\infty} \text{ is a } \delta\text{-covering of } A \Big\}.$$

It is not hard to verify that $\mathcal{H}^s(A)$ is nonincreasing with respect to s. The Hausdorff dimension of set A is formally defined to be

$$\dim_{\mathrm{H}}(A) = \inf\{t \in \mathbb{R} \mid \mathcal{H}^t(A) = 0\}.$$

We say a subset $A \subseteq \mathbb{R}$ is a *fractal* if $0 < \dim_{\mathrm{H}}(A) < 1$; and a curve Γ on the two-dimensional plane is a *fractal* if $\dim_{\mathrm{H}}(\Gamma) > 1$.

The above definitions of Hausdorff measure and Hausdorff dimension are quite abstract and do not seem to provide a natural method for calculation. As a consequence, the computation of the Hausdorff dimension of a set is considered very difficult. In the following, we justify this intuition on fractal sets in the one-dimensional line $\mathbb{R}$ and fractal curves on the two-dimensional plane $\mathbb{R}^2$.

First recall that a set $S \subseteq \mathbb{R}$ is *recursively open* if there exist recursive functions $\phi, \psi : \mathbb{N} \to \mathbb{D}$ and $\theta : \mathbb{N} \to \mathbb{N}$ such that (i) $S = \bigcup_{n=0}^{\infty} (\phi(n), \psi(n))$, and (ii) $\mu(\bigcup_{k=\theta(n)}^{\infty} (\phi(k), \psi(k))) < 2^{-n}$; that is, it is r.e. open and we can approximate S recursively. A set $T \subseteq \mathbb{R}$ is *recursively closed* if $\mathbb{R} - T$ is recursively open.

For a subset of $\mathbb{R}$, we note that if its Hausdorff dimension is less than 1, then its Lebesgue measure is zero and hence recursively approximable. So, it is natural to consider *strongly* polynomial-time approximable sets.

Theorem 5.1. (Ko [9]) *There exists a recursively closed, strongly polynomial-time approximable set $S \subseteq \mathbb{R}$ such that $\dim_{\mathrm{H}} S$ is a nonrecursive real number between 0 and 1.*

This result can be extended to two-dimensional curves.

Theorem 5.2. (Ko [9]) *There exists a polynomial-time computable function $f : [0,1] \to \mathbb{R}^2$ that defines a simple curve Γ such that $\dim_{\mathrm{H}} \Gamma$ is a nonrecursive real number between 1 and 2.*

Proof Idea. We start with a left r.e., nonrecursive real number

$$\alpha = \sum_{k \in K} 4^{-k},$$

where K is an r.e., nonrecursive subset of $\mathbb{N}$. Similar to the proof of Theorem 4.2, we can find a polynomial-time computable sequence $\{\alpha_n\}$ of dyadic rationals that converges to α, with the property of $\alpha_n \le \alpha_{n+1} < \alpha$, for all $n \ge 0$.

Next, we construct a simple fractal curve Γ on $\mathbb{R}^2$ whose Hausdorff dimension is equal to $3/2$. This curve Γ is the limit of curves Γ_n. Γ_0 is a horizontal line segment of length 1, which is a basic line segment. The curve Γ_n consists of 8^n basic line segments, each of length 4^{-n}. To get curve Γ_{n+1}, we replace each basic line segment of Γ_n by a basic curve that consists of 8 shorter basic line segments (see Figure 2).

To prove the theorem, we modify the curves Γ_n to Λ_n as follows: For each n, define a number p_n such that $2^n < p_n \le 2^{n+1}$. Let $\Lambda_1 = \Gamma_1$.

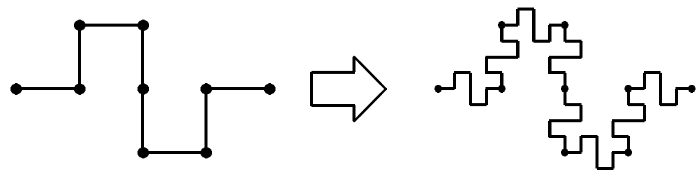

Figure 2. A basic curve of Γ_n changing to 8 basic curves of Γ_{n+1}.

Assume that the curve Λ_{2^n} has been defined, with m_{2^n} basic line segments. For all k such that $2^n < k \leq 2^n + p_n$, we replace each basic line segment of Λ_{k-1} by a basic curve as shown in Figure 2. For all k such that $2^n + p_n < k \leq 2^{n+1}$, we let $\Lambda_k = \Lambda_{k-1}$ (i.e., each basic line segment of Λ_{k-1} becomes 4 basic line segments of Λ_k). By choosing suitable values for p_n (depending on α_n and α_{n+1}), we can make m_{2^n} to be exactly $8^{2^n(1-\alpha_n)}$.

Now, we claim that the limit curve Λ of Λ_n has $\dim_{\mathrm{H}}(\Lambda) = (3/2)(1-\alpha)$, which is a nonrecursive real number. Since the curve Λ is the limit of simple curves Λ_n, it can be seen that the Hausdorff dimension of Λ is equal to its *box-counting dimension* (see Peitgen et al. [12] and Falconer [5]). To cover Λ with $4^{-k} \times 4^{-k}$ boxes, we only need m_k boxes (each of the diamond shape $\diamond$ with a basic line segment of Λ_k as one of its diagonals). Thus, the box-counting dimension of Λ is equal to

$$\lim_{k\to\infty} \frac{\log m_k}{-\log 4^{-k}} = \lim_{n\to\infty} \frac{\log 8^{2^n(1-\alpha_n)}}{-\log 4^{-2^n}}$$
$$= \lim_{n\to\infty} \frac{3 \cdot 2^n(1-\alpha_n)}{2 \cdot 2^n} = \frac{3}{2} \cdot (1-\alpha). \qquad \square$$

Intuitively, the Hausdorff dimension provides a measure of the relative density of a fractal set. Among all sets of the same Hausdorff dimension, Hausdorff measure provides a finer comparison of their *size*. For a set $A \subseteq \mathbb{R}$ with the Hausdorff dimension equal to one, its Hausdorff measure (with respect to dimension one) is equal to a computable constant times its Lebesgue measure. Therefore, if the Hausdorff dimension of a recursively approximable set A is one, then its one-dimensional Hausdorff measure is either infinite or is a finite recursive real number. However, when the Hausdorff dimension s of A is a fraction, then the s-dimensional Hausdorff measure of A may not be computable.

Theorem 5.3. (Ko [9]) *There exists a recursively closed, strongly polynomial-time approximable set* $T \subseteq [0,1]$ *such that* $\dim_{\mathrm{H}} T = 1/2$ *and* *its* $(1/2)$*-dimensional Hausdorff measure* $\mathcal{H}^{1/2}(T)$ *is a finite, nonrecursive real number.*

6. Julia Sets

An important group of fractals is generated by dynamical systems. In this section, we consider Julia sets of computable real functions f defined on the one-dimensional real line. Let $f:\mathbb{R}\to\mathbb{R}$ be a real-valued function. We let f^n denote the iteration of f for n times; i.e., $f^0(x)=x$ and $f^{n+1}(x)=f(f^n(x))$. We define the *(generalized) Julia set* of $f:\mathbb{R}\to\mathbb{R}$ to be

$$J(f)=\{x\in\mathbb{R}\mid(\exists B\in\mathbb{R})\,(\forall n)\;|f^n(x)|\le B\}.$$

Cenzer [2] gave a characterization of Julia sets of computable one-dimensional functions.

Theorem 6.1. (Cenzer [2]) *If* $A\subseteq\mathbb{R}$ *is a bounded, r.e. closed set with either a computable maximum point or a computable minimum point, then there is a polynomial-time computable function* g *such that* $J(g)=A$.

This characterization shows that most r.e. closed sets can be Julia sets and, hence, from the results of Section 4, that there exist easily computable functions f whose Julia sets are hard to compute.

Corollary 6.1. *(a) There exists a polynomial-time computable function* $f:\mathbb{R}\to\mathbb{R}$ *such that* $J(f)$ *is not recursively approximable.*

(b) There exists a polynomial-time computable function $f:\mathbb{R}\to\mathbb{R}$ *such that the Hausdorff dimension of* $J(f)$ *is not a recursive real number.*

It is well known that, for any bounded r.e. closed set A, there exists a computable function h such that $h(x)\ge 0$ for all $x\in\mathbb{R}$ and $A=\{x\in\mathbb{R}\mid h(x)=0\}$. The main idea of the proof of Cenzer's theorem is to modify this function into a new function g so that if $h(x)>0$ then $\lim_{n\to\infty}g^n(x)=\infty$ and hence $x\notin J(g)$ and if $h(x)=0$ then $g^k(x)=0$ and $x\in J(g)$. Thus, deciding whether a point x is in $J(g)$ or not is equivalent to deciding whether $h(x)>0$, and the theorem follows from the fact that deciding whether two given Cauchy functions represent two distinct real numbers or not is undecidable (see Ko [6]).

Intuitively, we suspect that a Julia set of a computable function f is difficult to compute because of the unpredictability of the behavior of $f^k(x)$ after a large number of iterations, rather than the inability to determine whether $f(x)>0$. To understand more clearly why Julia sets are hard to compute, we show in the following that a dynamical system can actually simulate the computation of a universal Turing machine M such that its Julia set corresponds to the complement of the halting set of M.

Theorem 6.2. (Ko [9]) *Let M be a Turing machine over the alphabet Σ. There exist polynomial-time computable functions $f : [0,1] \to \mathbb{R}$ and $h : \Sigma^* \to \mathbb{R}$ such that, for any $w \in \Sigma^*$ of length n,*
(i) if M accepts w, then $[h(w), h(w) + c^{-n}] \cap J(f) = \emptyset$, and
(ii) if M does not accept w, then $[h(w), h(w) + c^{-n}] \subseteq J(f)$,
where c is a positive constant.

Sketch of Proof. Basically, we treat the computation of M as a *discrete dynamical system*. That is, the mapping from a machine configuration to its successor configuration is a polynomial-time computable function g, and the iteration of this function is a dynamical system whose behavior is equivalent to the computation of the Turing machine M. The function f in the theorem is, thus, just a continuous function that encodes the discrete function g, and h a function that on input w encodes the halting configuration of M on w. □

Corollary 6.2. (Ko [9]) *There exists a polynomial-time comuptable function $f : \mathbb{R} \to \mathbb{R}$ such that $J(f)$ is not recursively approximable. In addition, f satisfies the following property: for any two dyadic rationals d, e, the question of whether $f(d) > e$ is decidable in polynomial time.*

Finally, we remark that there are studies of Julia sets of some specific dynamical systems. For instance, Rettinger and Weihrauch [14] have studied the Julia sets defined by the one-complex-variable functions of the form $f(z) = z^2 + c$. Other related results can be found in Saupe [16] and Zhong [18].

References

1. Barnsley, M. F., *Fractals Everywhere*, Academic Press, 2nd Edition, Boston, 1993.
2. Cenzer, D., Effective real dynamics, in *Logical Methods*, J. N. Crossley, J. B. Remmel, R.A. Shore, M. E. Sweedler, eds., Birkhäuser, Boston, 162–177, 1993.
3. Chou, A. W. and Ko, K., Computational complexity of two-dimensional regions, *SIAM J. Comput.* **24**, 923–947 (1995).
4. Du, D.-Z. and Ko, K., *Theory of Computational Complexity*, Wiley, New York, 2000.

5. Falconer, K., *Fractal Geometry, Mathematical Foundations and Applications*, Wiley, New York, 1991.
6. Ko, K., *Complexity Theory of Real Functions*, Birkhäuser, Boston, 1991.
7. Ko, K., A polynomial-time computable curve whose interior has a nonrecursive measure, *Theoret. Comput. Sci.* **145**, 241–270 (1995).
8. Ko, K., Polynomial-time computability in analysis, in *Handbook of Recursive Mathematics, Vol. 2: Recursive Algebra, Analysis and Combinatorics*, Yu. L. Ershov, S.S. Goncharov, A. Nerode and J. Remmel, eds., Studies in Logic and the Foundations of Mathematics, Vol. 139, Elsevier, Amsterdam, pp. 1271–1317, 1998.
9. Ko, K., On the computability of fractal dimensions and Hausdorff measure, *Annals of Pure and Appl. Logic* **93**, 195–216 (1998).
10. Ko, K. and Weihrauch, K., On the measure of two-dimensional regions with polynomial-time computable boundaries, in *Proceedings of 11th IEEE Conference on Computational Complexity*, IEEE Computer Society Press, 150–159, 1996.
11. Mandelbrot, B., *The Fractal Geometry of Nature*, W. H. Freeman, New York, 1983.
12. Peitgen, H.-O., Júrgens, H. and Saupe, D., *Chaos and Fractals, New Frontiers of Science*, Springer-Verlag, New York, 1992.
13. Pour-El, M. and Richards, I., *Computability in Analysis and Physics*, Springer-Verlag, Berlin, 1989.
14. Rettinger, R. and Weihrauch, K., The computational complexity of some Julia sets, in *Proceedings of the Fifth Workshop on Computability and Complexity in Analysis*, V. Brattka, M. Schröder, and K. Weihrauch, eds., FernUniversität, Hagen, 159–169, 2002.
15. Šanin, N., *Constructive Real Numbers and Function Spaces*, Translations of Mathematical Monographs, **21**, translated by E. Mendelson, American Mathematical Society, Providence, RI, 1968.
16. Saupe, D., Efficient computation of Julia sets and their fractal dimension, *Physica* **28D**, 358–370 (1987).
17. Weihrauch, K., *Computable Analysis, An Introduction*, Springer-Verlag, Berlin, 2000.
18. Zhong, N., Recursively enumerable subsets of $\mathbb{R}^q$ in two computing models, Blum-Shub-Smale machine and Turing machine, *Theoret. Comput. Sci.* **197**, 79–94 (1998).