

The Computational Complexity of Distance Functions of Two-Dimensional Domains

Arthur W. Chou* Ker-I Ko†

Abstract

We study the computational complexity of the distance function associated with a polynomial-time computable two-dimensional domains, in the context of the Turing machine-based complexity theory of real functions. It is proved that the distance function is not necessarily computable even if a two-dimensional domain is polynomial-time recognizable. On the other hand, if both the domain and its complement are strongly polynomial-time recognizable, then the distance function is polynomial-time computable if and only if $P = NP$.

1 Introduction

Assume that $S \subseteq \mathbb{R}^2$ is a bounded two-dimensional domain (i.e., a bounded, connected open set). We let $\delta_S(\mathbf{x})$ denote the distance between a point \mathbf{x} in \mathbb{R}^2 and the boundary Γ_S of set S . Intuitively, the distance function δ_S is computable if the set S itself is computable: We can search for the nearest point $\mathbf{y} \notin S$ and output the distance between \mathbf{x} and \mathbf{y} . Indeed, Brattka and Weihrauch [1999] showed that for several formulations of computable closed sets in \mathbb{R}^2 , the associated distance function is also computable.

When we consider the computational complexity of the distance function δ_S with respect to the computational complexity of the set S , the situation is different. For instance, in the context of the Turing machine-based complexity theory, Chou and Ko [1995] showed the following result: If $P \neq NP$, then there exists a simply connected domain $S \subseteq [0, 1]^2$ whose boundary Γ_S is a polynomial-time computable Jordan curve (i.e., the image of a polynomial-time computable function f from $[0, 1]$ to $[0, 1]^2$, which is one-to-one except that $f(0) = f(1)$), but its distance function δ_S is not polynomial-time computable.

In this note, we continue the investigation of the computational complexity of the distance functions δ_S of polynomial-time computable sets $S \subseteq [0, 1]^2$. We consider the following two formulations of polynomial-time computable two-dimensional regions [Chou and Ko, 1995]: A bounded two-dimensional domain S is called *polynomial-time recognizable* if there is a polynomial-time oracle Turing machine M such that, for any oracles ϕ_1, ϕ_2 representing a point $\mathbf{x} \in \mathbb{R}^2$ and any input integer $n > 0$, $M^{\phi_1, \phi_2}(n)$ correctly determines whether $x \in S$

*Department of Mathematics and Computer Science, Clark University, Worcester, MA 01610; e-mail: achou@vax.clarku.edu

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

for all points \mathbf{x} which have distance at least 2^{-n} away from the boundary of S . It is called *strongly polynomial-time recognizable* if, furthermore, $M^{\phi_1, \phi_2}(n)$ gives correct answers for all $\mathbf{x} \in S$ (thus, $M^{\phi_1, \phi_2}(n)$ can make mistakes only for those \mathbf{x} not in S but are within the distance of 2^{-n} of the boundary of S). The general question we ask is the following: What is the time complexity of δ_S if S is known to be polynomial-time recognizable, or strongly polynomial-time recognizable? Our main results can be summarized as follows:

(1) A polynomial-time recognizable two-dimensional domain S may have a non-computable distance function δ_S , even if S is simply connected and its boundary is a Jordan curve.

(2) If both a bounded, simply connected two-dimensional domain and its complement are strongly polynomial-time recognizable, then the associated distance function must be polynomial-time computable relative to a set in NP .

(3) If $P \neq NP$, then there exists a bounded, simply connected two-dimensional domain S whose boundary is a Jordan curve such that both S and its complement are strongly polynomial-time recognizable, but the associated distance function δ_S is not polynomial-time computable.

The above result (1) seems to suggest that the notion of polynomial-time recognizability is too weak compared with other notions of computable two-dimensional sets. Results (2) and (3) agree with earlier results of Chou and Ko [1995], and indicate that nondeterministic polynomial-time is the inherent complexity of distance functions.

Our basic computational model for real-valued functions and two-dimensional domains is the oracle Turing machine. For the theory of computational complexity of real functions based on this computational model, see Ko [1991, 1998] and Chou and Ko [1995]. We include a short summary of the definitions and notation of this theory in Section 2. For the general theory of computable analysis based on the Turing machine model, see, for instance, Pour-El and Richards [1989] and Weihrauch [2000]. The complexity classes defined in this paper are the standard ones of the discrete theory of NP -completeness; see, for instance, Du and Ko [2000].

2 Definitions and Notation

The basic computational objects in continuous computation are dyadic rationals $\mathbb{D} = \{m/2^n : m \in \mathbb{Z}, n \in \mathbb{N}\}$. Each dyadic rational d has infinitely many binary representations, with arbitrarily many trailing zeros. For each $n \in \mathbb{N}$, we let \mathbb{D}_n denote the class of dyadic rationals which have a binary representation of at most n bits to the right of the binary point; that is, $\mathbb{D}_n = \{m/2^n : m \in \mathbb{Z}\}$.

We say a function $\phi : \mathbb{N} \rightarrow \mathbb{D}$ *binary converges* to a real number x , or *represents a real number x* , if (i) for all $n \geq 0$, $\phi(n) \in \mathbb{D}_n$, and (ii) for all $n \geq 0$, $|x - \phi(n)| \leq 2^{-n}$. For any $x \in \mathbb{R}$, there is a unique function $\phi_x : \mathbb{N} \rightarrow \mathbb{D}$ that binary converges to x and satisfies the condition $x - 2^{-n} < \phi_x(n) \leq x$ for all $n \geq 0$. We call this function ϕ_x the *standard Cauchy function* for x .

To compute a real-valued function $f : \mathbb{R} \rightarrow \mathbb{R}$, we use oracle Turing machines (TM's) as the computational model. We say an oracle Turing machine M *computes* a function $f : \mathbb{R} \rightarrow \mathbb{R}$ if, for a given oracle ϕ that binary converges to a real number x and for a given input $n > 0$, $M^\phi(n)$ halts and outputs a dyadic rational e such that $|e - f(x)| \leq 2^{-n}$. When

the oracle ϕ is the standard Cauchy function for x , we also write $M^x(n)$ to denote the computation of $M^\phi(n)$. We say a function $f : \mathbb{R} \rightarrow \mathbb{R}$ is *polynomial-time computable* if there exists a polynomial-time oracle Turing machine that computes f .

We write \mathbf{x} or $\langle x_1, x_2 \rangle$, where $x_1, x_2 \in \mathbb{R}$, to denote a point in the two-dimensional plane \mathbb{R}^2 . For any two points $\mathbf{x} = \langle x_1, x_2 \rangle$ and $\mathbf{y} = \langle y_1, y_2 \rangle$ in \mathbb{R}^2 , we write $\text{dist}(\mathbf{x}, \mathbf{y})$ or $|\mathbf{x} - \mathbf{y}|$ to denote the distance $\sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$ between them. For any point $\mathbf{x} \in \mathbb{R}^2$ and a closed set $A \subseteq \mathbb{R}^2$, we write $\text{dist}(\mathbf{x}, A) = \text{dist}(A, \mathbf{x}) = \min\{\text{dist}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in A\}$. For any domain $S \subseteq \mathbb{R}^2$, let $\delta_S(\mathbf{x}) = \text{dist}(\mathbf{x}, \Gamma_S)$, where Γ_S is the boundary of S .

The notions of computable and polynomial-time computable real functions can be extended naturally to functions $f : \mathbb{R} \rightarrow \mathbb{R}^2$ and functions $f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$. In particular, when an element of the domain of the function f is a point $\langle x_1, x_2 \rangle$ in \mathbb{R}^2 , the corresponding oracle Turing machine uses two oracles ϕ_1, ϕ_2 which binary converge to x_1 and x_2 , respectively.

For any set $S \subseteq \mathbb{R}^2$, let χ_S denote its characteristic function; i.e., $\chi_S(\mathbf{x}) = 1$ if $\mathbf{x} \in S$, and $\chi_S(\mathbf{x}) = 0$ otherwise. Intuitively, S is computable (or, polynomial-time computable) if the function χ_S is computable (or, respectively, polynomial-time computable). Since χ_S is discontinuous at the boundary of S , the definition based on this concept is too strict. That is, suppose that we define a set S to be polynomial-time computable if there is a polynomial time oracle Turing machine computing χ_S ; then, only two trivial sets, \mathbb{R}^2 and \emptyset , are polynomial-time computable. Chou and Ko [1995] considered two different ways to relax the computability requirements of this concept. One of them is the following:

Definition 2.1 (a) *A set $S \subseteq \mathbb{R}^2$ is called polynomial-time recognizable if there exist an oracle TM M and a polynomial p such that $M^{\phi, \psi}(n)$ computes $\chi_S(\mathbf{z})$ in time $p(n)$ whenever (ϕ, ψ) represents a point \mathbf{z} in \mathbb{R}^2 whose distance to the boundary Γ_S of S is greater than 2^{-n} ; i.e., the error set*

$$E_n(M) = \{\mathbf{z} \in \mathbb{R}^2 : (\exists (\phi, \psi) \text{ representing } \mathbf{z}) [M^{\phi, \psi}(n) \neq \chi_S(\mathbf{z})]\}$$

is a subset of $\{\mathbf{z} \in \mathbb{R}^2 : \text{dist}(\mathbf{z}, \Gamma_S) \leq 2^{-n}\}$.

(b) *A set $S \subseteq \mathbb{R}^2$ is called strongly polynomial-time recognizable if there exist an oracle TM M and a polynomial p which satisfy the conditions of (a) above and, in addition, $E_n(M) \cap S = \emptyset$.*

We note that if both S and its complement $S^c = \mathbb{R}^2 - S$ are strongly polynomial-time recognizable, then we can combine the two underlying machine to determine, for any point \mathbf{x} , whether it is in S or is in S^c , or is within distance 2^{-n} of the boundary. This provides a stronger notion of polynomial-time computability of two-dimensional domains.

3 Distance Function of a Polynomial-Time Recognizable Set

In this section, we show that polynomial-time recognizability of a two-dimensional domain S does not warrant even the computability of the associated distance function. We first show a simple example in which the boundary of the set S is not a Jordan curve.

Theorem 3.1 *For any real number $r \in (0, 1/2)$, there exists a bounded, simply connected open set $S \subseteq [0, 1]^2$ such that S is polynomial-time recognizable, but $\delta_S(\langle 1/2, 1/2 \rangle) = r$.*

Proof. Let $s = 1/2 - r$. Let L denote the line segment from $\langle 0, 1/2 \rangle$ to $\langle s, 1/2 \rangle$. Define

$$S = (0, 1)^2 - L.$$

It is clear that $\delta_S(\langle 1/2, 1/2 \rangle) = 1/2 - s = r$. We claim that S is polynomial-time recognizable. Indeed, as far as polynomial-time recognizability is concerned, there is no difference between set S and $[0, 1]^2$. An oracle TM for S can determine whether a point \mathbf{x} represented by oracles (ϕ_1, ϕ_2) is in S or not by checking whether an approximate dyadic point \mathbf{d} of \mathbf{x} , given by the oracle, is in $[0, 1]^2$ or not. All the errors occur only near the boundary of the square $[0, 1]^2$ or on the line segment L . \square

In the above example, the distance $\delta_S(\langle 1/2, 1/2 \rangle)$ could be an arbitrary real number in $(0, 1/2)$. This seems due to the fact that the boundary of set S is not a Jordan curve, and hence the Turing machine M that recognizes S can essentially ignore the line segment L . Indeed, if we require that the boundary Γ_S be a Jordan curve then, for any computable point $\mathbf{x} \in [0, 1]^2$, $\delta_S(\mathbf{x})$ cannot be an arbitrary real number any more, though it may still be a non-computable real number.

We say that a real number r is a *right r.e.* real number if its right cut $R_r = \{d \in \mathbb{D} : d > r\}$ is an r.e. set. This means that there exists a Turing Machine M_1 which enumerates the set $R_r \cap (0, 1)$; i.e., M_1 prints strings representing dyadic rationals d in $R_r \cap (0, 1)$ one by one on its output tape. Similarly, we say that s is a *left r.e.* real number if its left cut $L_s = \{d \in \mathbb{D} : d < s\}$ is an r.e. set. We refer to Ko [1983, 1991] for some basic discussions of these notions. (Note that in Downey [2001] and Zheng [2002] “right r.e.” real numbers are called “r.e.” real numbers or “left computable”, and that “left r.e.” real numbers are called “co-r.e.” or “right computable”.)

Theorem 3.2 *Let $S \subseteq [0, 1]^2$ be a simply connected open set whose boundary Γ_S is a Jordan curve. If S is polynomial-time recognizable, then for every computable point $\mathbf{x} \in [0, 1]^2$, $\delta_S(\mathbf{x})$ must be a right r.e. real number.*

Proof. Let $T = \mathbb{R}^2 - (S \cup \Gamma_S)$. Let \mathbf{x} be a fixed computable point in $[0, 1]^2$. Then, there is a computable sequence $\{\mathbf{x}_n\}$ of dyadic rational points in $[0, 1]^2$ that binary converges to \mathbf{x} (thus, $|\mathbf{x}_n - \mathbf{x}| \leq 2^{-n}$). Let $r = \delta_S(\mathbf{x})$. Assume that M_1 is a TM that polynomial-time recognizes set S . Consider the following TM M that halts on dyadic rationals d in the right cut of r :

Input: $d \in \mathbb{D}$.
 For $m := 1$ to ∞ do
 For $\mathbf{e} = \langle e_1, e_2 \rangle \in (\mathbb{D}_{m+2})^2 \cap [0, 1]^2$ do
 Simulate $M_1^{e_1, e_2}(m+2)$;
 If $M_1^{e_1, e_2}(m+2) = 0$ and $|\mathbf{x}_{m+2} - \mathbf{e}| \leq d - 2^{-m}$ then halt;

First, assume that $d > r = \delta_S(\mathbf{x})$. Then, there exists a point \mathbf{y} in Γ_S such that $|\mathbf{x} - \mathbf{y}| = r$. Since Γ_S is a Jordan curve, any open neighborhood of \mathbf{y} must contain a point in T ; furthermore, it must contain a dyadic rational point in T , since \mathbb{D}^2 is dense in \mathbb{R}^2 . Let k be the least integer such that

- (i) there exists a point $\mathbf{e} \in (\mathbb{D}_{k+2})^2 \cap T$ such that $|\mathbf{e} - \mathbf{y}| \leq 2^{-(k+2)}$, and

(ii) $d - 2^{-k} > r$.

Fix a point $\mathbf{e} = \langle e_1, e_2 \rangle$ satisfying condition (i), and let j be the least integer such that

(iii) $\delta_S(\mathbf{e}) \geq 2^{-j}$.

Let $m = \max\{k, j\} + 1$.

We claim that M will halt in the m th iteration if it did not halt before. In the m th iteration, when \mathbf{e} is equal to the above fixed point, from condition (iii), $M_1^{e_1, e_2}(m+2)$ must output 0. In addition, we have

$$\begin{aligned} |\mathbf{x}_{m+2} - \mathbf{e}| &\leq |\mathbf{x}_{m+2} - \mathbf{x}| + |\mathbf{x} - \mathbf{y}| + |\mathbf{y} - \mathbf{e}| \\ &\leq 2^{-(m+2)} + r + 2^{-(k+2)} < d - 2^{-k} + 2^{-(k+2)} + 2^{-(m+2)} \\ &\leq d - 2^{-(k+1)} \leq d - 2^{-m}. \end{aligned}$$

Therefore, M will halt at this step.

Conversely, assume that M halts on input d with respect to integer m and point $\mathbf{e} = \langle e_1, e_2 \rangle$. Since $M_1^{e_1, e_2}(m+2) = 0$, we have either $\mathbf{e} \in T$ or $\delta_S(\mathbf{e}) \leq 2^{-(m+2)}$. In either case, we have

$$\begin{aligned} \delta_S(\mathbf{x}) &\leq |\mathbf{x} - \mathbf{e}| + 2^{-(m+2)} \\ &\leq |\mathbf{x} - \mathbf{x}_{m+2}| + |\mathbf{x}_{m+2} - \mathbf{e}| + 2^{-(m+2)} \\ &\leq 2^{-(m+2)} + d - 2^{-m} + 2^{-(m+2)} = d - 2^{-(m+1)} < d. \end{aligned}$$

Therefore, M works correctly on d . □

Theorem 3.3 *For any right r.e. real number $r \in (0, 1/2)$, there is a simply connected open set $S \subseteq [0, 1]^2$ whose boundary Γ_S is a Jordan curve such that S is polynomial-time recognizable and $\delta_S(\langle 1/2, 1/2 \rangle) = r$.*

Proof. Let $s = 1/2 - r$. Then, s is left r.e.; i.e., its left cut $L_s = \{d \in \mathbb{D} : d < s\}$ is an r.e. set. This means that there exists a TM M_1 that enumerates the set $L_s \cap (0, 1)$; i.e., M_1 prints strings representing dyadic rationals d in $L_s \cap (0, 1)$ one by one on its output tape. Let s_1 be the first dyadic rational printed by M_1 , and, for $n > 1$, $s_n = \max(\{d \in \mathbb{D} : M_1 \text{ prints } d \text{ within } n \text{ moves}\} \cup \{s_1\})$. It is apparent that $s_1 \leq s_2 \leq \dots$, and $\lim_{n \rightarrow \infty} s_n = s$. In addition, the sequence $\{s_n\}_{n=1}^{\infty}$ is polynomial-time computable.

Now, define rectangles S_n recursively as follows:

- (i) S_1 is the rectangle of width s_1 and height 2^{-2} , whose upper left corner is $\langle 0, 1/2 \rangle$.
- (ii) For $n \geq 2$, if $s_n = s_{n-1}$, then $S_{n-1} = S_n$.
- (iii) If $n \geq 2$ and $s_n > s_{n-1}$, then S_n is the rectangle of width $s_n - s_{n-1}$ and height $2^{-(n+1)}$, whose upper left corner is $\langle s_{n-1}, 1/2 \rangle$ (i.e., the upper left corner of S_n is the same as the upper right corner of S_{n-1}).

Define

$$S = (0, 1)^2 - \bigcup_{n=1}^{\infty} S_n.$$

It is clear that $\delta_S((1/2, 1/2)) = 1/2 - s = r$. Since $\lim_{n \rightarrow \infty} s_n = s$, it follows that the boundary of S is a Jordan curve.

To see that S is polynomial-time recognizable, consider the following oracle Turing machine M :

Oracles: (ϕ_1, ϕ_2) , representing a point $\mathbf{x} \in \mathbb{R}^2$.

Input: $n > 0$.

- (1) Ask the oracles to get a dyadic rational point $\mathbf{d} \in \mathbb{R}^2$ such that $|\mathbf{d} - \mathbf{x}| \leq 2^{-n}$.
- (2) Compute s_1, s_2, \dots, s_n , and construct S_1, \dots, S_n .
- (3) If $\mathbf{d} \notin [0, 1]^2$ or if $\mathbf{d} \in \bigcup_{i=1}^n S_i$, then output 0, else output 1.

Without loss of generality, assume that both \mathbf{x} and \mathbf{d} are in $[0, 1]^2$. Then, the answer given by M can be wrong only if (a) $\mathbf{d} \notin \bigcup_{i=1}^n S_i$ but $\mathbf{x} \in \bigcup_{i=1}^n S_i$, or (b) $\mathbf{d} \in \bigcup_{i=1}^n S_i$ but $\mathbf{x} \notin \bigcup_{i=1}^n S_i$, or (c) $\mathbf{x} \in S_k$ for some $k > n$ with $s_k > s_{k-1} \geq s_n$. In cases (a) and (b), \mathbf{x} and \mathbf{d} lie in the opposite sides of the boundary Γ_S and so \mathbf{x} is within distance 2^{-n} of the boundary. In case (c), the condition $s_k > s_n$ implies that S_k is different from S_n and the height of S_k is $2^{-(k+1)} < 2^{-n}$, and so \mathbf{x} must be within distance 2^{-n} of the boundary Γ_S . Therefore, M recognizes set S . \square

Corollary 3.4 *There exists a simply connected open set $S \subseteq [0, 1]^2$ whose boundary Γ_S is a Jordan curve such that S is polynomial-time recognizable and δ_S is not a computable real function.*

Proof. A computable real function must map a computable point \mathbf{x} to a computable real number. It is known (see, e.g., Ko [1991]) that there are right r.e. real numbers which are not computable. \square

4 Distance Function of a Strongly Polynomial-Time Recognizable Set

We have seen, in the last section, that for a polynomial-time recognizable set S , the distance function may not even be computable. In this section, we consider sets S with the property that both S and its complement S^c are strongly polynomial-time recognizable. For such sets, we show that the associated distance functions are polynomial-time computable if and only if $P = NP$.

Recall that P is the class of sets (of binary strings) that are acceptable by polynomial-time deterministic TM's, and NP is the class of sets (of binary strings) that are acceptable by polynomial-time nondeterministic TM's.

Theorem 4.1 *Assume that $S \subseteq [0, 1]^2$ is a simply connected open set. If both S and $S^c = \mathbb{R}^2 - S$ are strongly polynomial-time recognizable, then δ_S is polynomial-time computable relative to an oracle set $A \in NP$.*

Proof. Let M_1 and M_0 be the oracle TM's that strongly polynomial-time recognize sets S and S^c , respectively. Let $p(n)$ be a polynomial function that bounds the running time of both M_1 and M_0 . Define

$$A = \{\langle d_1, d_2, L, n, i \rangle : d_1, d_2, L \in \mathbb{D}_n, n \geq 1, i \in \{0, 1\}, \\ (\exists e_1, e_2 \in \mathbb{D}_{p(n)})[M_i^{e_1, e_2}(n) = 1, |\langle d_1, d_2 \rangle - \langle e_1, e_2 \rangle| \leq L]\}.$$

It follows immediately from the existential quantifier characterization of NP (see, e.g., Du and Ko [2000]) that A is in NP . The following TM M computes δ_S using oracle A .

Oracles: Set A ; functions ϕ_1, ϕ_2 representing a point $\mathbf{x} \in \mathbb{R}^2$. (Without loss of generality, assume that $\mathbf{x} \in [0, 1]^2$.)

Input: $n > 0$.

- (1) Ask oracles ϕ_1, ϕ_2 to find a point $\mathbf{d} = \langle d_1, d_2 \rangle \in (\mathbb{D}_{n+1})^2$ such that $|\mathbf{d} - \mathbf{x}| \leq 2^{-(n+1)}$.
- (2) Simulate M_1 and M_0 to get $a = M_1^{\phi_1, \phi_2}(n+1)$ and $b = M_0^{\phi_1, \phi_2}(n+1)$.
- (3) If $a = 1$ and $b = 0$, then binary search for $L \in \mathbb{D}_{n+1} \cap [0, 2]$ such that $\langle d_1, d_2, L, n+1, 0 \rangle \in A$ but $\langle d_1, d_2, L + 2^{-(n+1)}, n+1, 0 \rangle \notin A$; output L .
- (4) If $a = 0$ and $b = 1$, then binary search for $L \in \mathbb{D}_{n+1} \cap [0, 2]$ such that $\langle d_1, d_2, L, n+1, 1 \rangle \in A$ but $\langle d_1, d_2, L + 2^{-(n+1)}, n+1, 1 \rangle \notin A$; output L .
- (5) If $a = 1$ and $b = 1$, then output 0.

First, we note that for any \mathbf{x} , the simulation of step (2) cannot output $a = b = 0$, since \mathbf{x} is either in S or in S^c . Thus, the above algorithm for machine M is well defined.

Next, we verify that machine M computes δ_S correctly. If M reaches step (5), then one of M_1 or M_0 must have made a mistake. That means \mathbf{x} must be within distance $2^{-(n+1)}$ of the boundary Γ_S of S . So, the output 0 is correct within error $2^{-(n+1)}$.

Assume that M reaches step (3). Then, we must have $\mathbf{x} \in S$. Suppose M outputs L . Then, we have $\langle d_1, d_2, L, n+1, 0 \rangle \in A$, which implies that there exists a point $\mathbf{e} = \langle e_1, e_2 \rangle$ in $(\mathbb{D}_{p(n+1)})^2$ such that $M_0^{e_1, e_2}(n+1) = 1$ and $|\mathbf{e} - \mathbf{d}| \leq L$. From $M_0^{e_1, e_2}(n+1) = 1$, we know that either $\mathbf{e} \in S^c$ or $\delta_S(\mathbf{e}) \leq 2^{-(n+1)}$. Either way, we get

$$\delta_S(\mathbf{x}) \leq |\mathbf{x} - \mathbf{d}| + |\mathbf{d} - \mathbf{e}| + 2^{-(n+1)} \leq L + 2^{-n}.$$

On the other hand, let \mathbf{y} be any point in Γ_S . Then, for the standard Cauchy functions ψ_1, ψ_2 for \mathbf{y} , we must have $M_0^{\psi_1, \psi_2}(n+1) = 1$. Let $e_1 = \psi_1(p(n+1))$ and $e_2 = \psi_2(p(n+1))$. We must also have $M_0^{e_1, e_2}(n+1) = 1$ because M_0 cannot distinguish between \mathbf{y} and $\mathbf{e} = \langle e_1, e_2 \rangle$ within $p(n+1)$ moves. Now, $\langle d_1, d_2, L + 2^{-(n+1)}, n+1, 0 \rangle \notin A$ implies that $|\mathbf{d} - \mathbf{e}| > L + 2^{-(n+1)}$; or

$$|\mathbf{x} - \mathbf{y}| \geq |\mathbf{d} - \mathbf{e}| - |\mathbf{x} - \mathbf{d}| - |\mathbf{y} - \mathbf{e}| > L - 2^{-(n+1)}.$$

Since \mathbf{y} is an arbitrary point in Γ_S , we get $\delta_S(\mathbf{x}) > L - 2^{-(n+1)}$. Together, we get $|L - \delta_S(\mathbf{x})| \leq 2^{-n}$.

The case of M reaching step (4) is similar to the above case. To be more precise, if M reaches step (4), we must have $\mathbf{x} \in S^c$. Suppose M outputs L . Then, using the same

argument, we can prove that $\langle d_1, d_2, L, n+1, 1 \rangle \in A$ implies $\delta_S(\mathbf{x}) \leq L + 2^{-n}$. For the second half of the proof, we note that for any point $\mathbf{z} \in \Gamma_S$, we can find a point $\mathbf{y} \in S$ with $|\mathbf{y} - \mathbf{z}| \leq 2^{-(n+1)}$. Now, using this point \mathbf{y} , we can show, by the same argument, that $\langle d_1, d_2, L + 2^{-(n+1)}, n+1, 1 \rangle \notin A$ implies $|\mathbf{x} - \mathbf{y}| > L - 2^{-(n+1)}$ and, hence, $|\mathbf{x} - \mathbf{z}| > L - 2^{-n}$. Together, we get $|L - \delta_S(\mathbf{x})| \leq 2^{-n}$.

Finally, we check that, in steps (3) and (4), the binary search needs to ask the oracles at most $n+2$ times, and so the machine M runs in polynomial time. Thus, δ_S is polynomial-time computable relative to an oracle in NP . \square

When the boundary Γ_S of set S is a Jordan curve, a Turing machine that strongly polynomial-time recognizes set $T = \mathbb{R}^2 - (S \cup \Gamma_S)$ works almost the same as one that strongly polynomial-time recognizes S^c . So, we get the following stronger result.

Corollary 4.2 *Assume that $S \subseteq [0, 1]^2$ is a simply connected open set whose boundary Γ_S is a Jordan curve. If both S and $T = \mathbb{R}^2 - (S \cup \Gamma_S)$ are strongly polynomial-time recognizable, then δ_S is polynomial-time computable relative to a set $A \in NP$.*

We note that the set S in the proof of Theorem 3.1 has the property that both S and $T = \mathbb{R}^2 - (S \cup \Gamma_S)$ are strongly polynomial-time recognizable. Thus, the condition in Corollary 4.2 that the boundary Γ_S is a Jordan curve is necessary.

Next, we show that the oracle set A in NP in Theorem 4.1 for the computation of δ_S is necessary.

Theorem 4.3 *Assume that $P \neq NP$. Then, there exists a simply connected open set $S \subseteq [0, 1]^2$ whose boundary Γ_S is a Jordan curve, such that both S and $T = \mathbb{R}^2 - (S \cup \Gamma_S)$ are strongly polynomial-time recognizable, but δ_S is not polynomial-time computable.*

Proof. Assume that $A \subseteq \{0, 1\}^*$ is a set in $NP - P$. Then, from the existential quantifier characterization of NP , we know that there exist a set $B \in P$ and a polynomial function p such that, for every string $w \in \{0, 1\}^*$ of length n ,

$$w \in A \iff (\exists u, |u| = p(n)) \langle w, u \rangle \in B.$$

For each string $t \in \{0, 1\}^*$ of length m , we write i_t to denote the unique integer between 0 and $2^m - 1$ whose m -bit binary expansion (with possible leading zeroes) is equal to t .

For each $n > 0$, let $a_n = 1 - 2^{-(n-1)}$. We divide the interval $[a_n, a_{n+1}]$ into 2^n subintervals of equal length, each corresponding to a string $w \in \{0, 1\}^n$. To be more precise, for each string $w \in \{0, 1\}^n$, we let $r_w = a_n + i_w \cdot 2^{-2n}$, and let $I_w = [r_w, r_w + 2^{-2n}]$. We further divide I_w into $2^{p(n)}$ subintervals of equal length, each corresponding to a string u of length $p(n)$. That is, for each string u of length $p(n)$, we let $s_{w,u} = r_w + i_u \cdot 2^{-p(n)-2n}$, and $J_{w,u} = [s_{w,u}, s_{w,u} + 2^{-p(n)-2n}]$. For each u of length $p(n)$, we also define

$$h_u = \begin{cases} (2^{p(n)-1} - i_u) \cdot 2^{-p(n)-2n} & \text{if } i_u < 2^{p(n)-1}, \\ (i_u - 2^{p(n)-1} + 1) \cdot 2^{-p(n)-2n} & \text{if } i_u \geq 2^{p(n)-1}. \end{cases}$$

Then, we define a rectangle $T_{w,u}$ as follows: the rectangle $T_{w,u}$ has width $2^{-p(n)-2n}$, height h_u , and its lower left corner is $\langle s_{w,u}, 0 \rangle$.

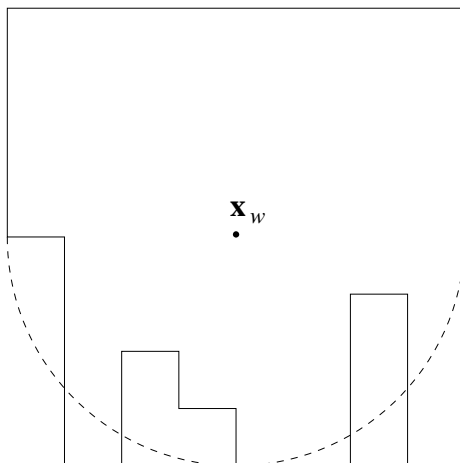


Figure 1: Set S within the square $I_w \times [0, 2^{-2n}]$.

Finally, define set

$$S = (0, 1)^2 - \bigcup_{\langle w, u \rangle \in B} T_{w, u}.$$

Figure 1 shows set $S \cap I_w \times [0, 2^{-2n}]$, when, for instance, $p(n) = 3$, and $\langle w, 000 \rangle$, $\langle w, 010 \rangle$, $\langle w, 011 \rangle$, $\langle w, 110 \rangle$ are the only pairs $\langle w, u \rangle$ in B . The above limiting process clearly shows that the boundary of S is a Jordan curve.

Define $\mathbf{x}_w = \langle r_w + 2^{-2n-1}, 2^{-2n-1} \rangle$. Then, we can see easily that if $w \notin A$, then $\delta_S(\mathbf{x}_w)$ is equal to 2^{-2n-1} . If $w \in A$, then we remove at least one $T_{w, u}$ from S and so $\delta_S(\mathbf{x}_w)$ is less than $2^{-2n-1} - 2^{-p(n)-2n-1}$ (cf. Figure 1). Thus, whether $w \in A$ can be determined from an approximation d to $\delta_S(\mathbf{x}_w)$ within error $2^{-p(n)-2n-3}$. This means that δ_S is not polynomial-time computable, since we assumed that $A \notin P$.

It is left to show that both sets S and $T = \mathbb{R}^2 - (S \cup \Gamma_S)$ are strongly polynomial-time recognizable. In the following, we show an oracle TM M that strongly polynomial-time recognizes set S . The machine for set T is similar, and we omit it. Let M_B be the TM that determines whether $\langle w, u \rangle \in B$ in polynomial time.

Oracles: ϕ_1, ϕ_2 representing a point $\mathbf{x} \in \mathbb{R}^2$.

Input: $n > 0$.

- (1) Let $d_1 = \phi_1(p(n) + 2n)$ and $d_2 = \phi_2(p(n) + 2n)$. If $d_1 \notin (0, 1)$, then output 0 and halt.
- (2) Find integer k such that $a_k \leq d_1 < a_{k+1}$. If $k > n$, then output 1 if and only if $0 < d_2 < 1$, and halt.
- (3) If $k \leq n$, then find $w, u \in \{0, 1\}^*$ of length n and $p(n)$, respectively, such that $d_1 \in J_{w, u}$.
- (4) Simulate M_B on $\langle w, u \rangle$. If $\langle w, u \rangle \notin B$, then output 1 if and only if $0 < d_2 < 1$; otherwise, output 1 if and only if $h_u < d_2 < 1$.

The correctness of the machine M is clear. In particular, if it gets $k > n$ in step (2), then we know that the line segment from $\langle a_k, 0 \rangle$ to $\langle 1, 0 \rangle$ is within distance 2^{-2n} of the

lower bottom of the boundary of S , and so the answer based on the condition $0 < d_2 < 1$ is either correct or incorrect but acceptable. We also observe that the computation of M runs obviously in polynomial time. Thus, S is strongly polynomial-time recognizable. \square

Corollary 4.4 *The following are equivalent:*

(a) $P = NP$.

(b) *For every simply connected open set $S \subseteq [0, 1]^2$, if both S and S^c are strongly polynomial-time recognizable, then δ_S is polynomial-time computable.*

(c) *For every simply connected open set $S \subseteq [0, 1]^2$ whose boundary is a Jordan curve, if both S and $T = \mathbb{R}^2 - (S \cup \Gamma_S)$ are strongly polynomial-time recognizable, then δ_S is polynomial-time computable.*

References

- Brattka, V. and Weihrauch, K. [1999], Computability on subsets of Euclidean space I: Closed and compact subsets, *Theoret. Comput. Sci.* **219**, 65–93.
- Chou, A. W. and Ko, K. [1995], Computational complexity of two-dimensional regions, *SIAM J. Comput.* **24**, 923–947.
- Chou, A. W. and Ko, K. [2004], On the complexity of finding paths in a two-dimensional domain I: shortest paths, *Math. Logic Quart.* **50**, 551–572; preliminary version in *Proc. International Conference on Computability and Complexity in Analysis*, 2003, Hagen, Germany.
- Downey, R. G. [2001], Some computability-theoretical aspects of real and randomness, Preprint, September 2001.
- Du, D.-Z. and Ko, K. [2000], *Theory of Computational Complexity*, John Wiley & Sons, New York.
- Ko, K. [1983], On the definitions of some complexity classes of real numbers, *Math. System Theory* **16**, 95–109.
- Ko, K. [1991], *Complexity Theory of Real Functions*, Birkhäuser, Boston.
- Ko, K. [1998], Polynomial-time computability in analysis, in *Handbook of Recursive Mathematics, Vol. 2: Recursive Algebra, Analysis and Combinatorics*, Yu. L. Ershov et al., eds., Studies in Logic and the Foundations of Mathematics, Vol. 139, Elsevier, Amsterdam, 1998, pp. 1271–1317.
- Pour-El, M. and Richards, I. [1989], *Computability in Analysis and Physics*, Springer-Verlag, Berlin.
- Weihrauch, K. [2000], *Computable Analysis*, Springer-Verlag, Heidelberg.
- Zheng, X. [2002], Recursive approximability of real numbers. *Mathematical Logic Quarterly*, **48** (Suppl. 1), 131–156.