Theoretical Computer Science 852 (2021) 1–17

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On continuous one-way functions

Ker-I Ko [a], Lidong Wu [b,*]

[a] State University of New York at Stony Brook, Stony Brook, NY 11794, United States of America
[b] University of Texas at Tyler, Tyler, TX 75799, United States of America

ARTICLE INFO

Article history:
Received 27 May 2020
Received in revised form 2 October 2020
Accepted 23 October 2020
Available online 1 December 2020

Keywords:
Continuous functions
Polynomial-time
Inverse modulus of continuity

ABSTRACT

The existence of one-way functions seems to depend, intuitively, on certain irregular properties of polynomial-time computable functions. Therefore, for functions with continuity properties, it suggests that all such functions are not one-way. It is shown here that in the formal complexity theory of real functions, this nonexistence of continuous one-way functions can be proved for one-to-one one-dimensional real functions, but fails for one-to-one two-dimensional real functions, if certain strong discrete one-way functions exist. Furthermore, for k-to-one functions, we can prove the existence of four-to-one one-dimensional one-way functions under the same assumption of the existence of strong discrete one-way functions. (A function f is k-to-one if for any y there exist at most k distinct values x such that $f(x) = y$.)

1. Introduction

The notion of one-way functions has played a central role in several areas of theoretical computer science, including cryptography and pseudorandom number generation. In these areas, certain strong one-way functions are often assumed, without a proof that these one-way functions actually exist. From the complexity-theoretic point of view, it is important to find the necessary and sufficient conditions for the existence of such one-way functions. For a certain weak form of one-way functions, such characterizations by the relations between complexity classes have been known. For example, let us call a function ϕ from finite strings to finite strings a *one-way function* if ϕ is one-to-one, polynomially honest (i.e., there exists a polynomial function p such that $p(l(\phi(s))) \geq l(s)$ for all inputs s, where $l(t)$ is the length of t) and polynomial-time computable but is not polynomial-time invertible (i.e., for any function ψ such that $\phi(\psi(t)) = t$ for all $t \in Range(\phi)$, ψ is not polynomial-time computable). Then, it is known [2,3] that such a one-way function exists if and only if $P \neq UP$, where UP is the class of sets computable in polynomial time by some unambiguous nondeterministic machines (see Section 2 for the formal definition of the class *UP*). In the above, the requirement that ϕ be polynomially honest is necessary in order to exclude trivial one-way functions whose inverses map some strings to exponentially long strings. In general, however, the requirement of one-to-oneness is not necessary. Let us call a function ϕ a k-to-one function if for any t, there are at most k strings $s_1, \cdots, s_k$ such that $\phi(s_i) = t$, $1 \leq i \leq k$. The existence of k-to-one one-way functions is often assumed in cryptography. The computational complexity of k-to-one one-way functions has also been studied in literature (see, for example, Watanabe [7]).

Intuitively, the inverse of a polynomial-time computable, polynomially honest function ϕ is difficult to compute because the function ϕ could be irregular in the sense that the values of $\phi(s)$ and $\phi(t)$ do not have any obvious relation even if

* Corresponding author.
E-mail addresses: lwu@uttyler.edu, wuld666@gmail.com (L. Wu).

https://doi.org/10.1016/j.tcs.2020.10.031
0304-3975/

s and t are nicely related. On the other hand, if the function ϕ does show some regularity, then the inverse ϕ^{-1} may be easy to compute. For example, if we know that ϕ is order-preserving in the sense that $s \le t$ implies $\phi(s) \le \phi(t)$ for some natural ordering $\le$, then a simple binary search procedure computes the inverse ϕ^{-1} easily. The above observation suggests that if ϕ is a *continuous* function, in the general sense that $\phi(s)$ and $\phi(t)$ satisfy certain relation whenever s and t satisfy some similar relation, then the inverse of ϕ could be easy to compute. For instance, in an informal setting, we can see easily that a one-to-one function f which maps real numbers in a closed interval $[a, b]$ to real numbers must preserve the natural ordering on reals and thus should have a polynomial-time computable inverse. It should be cautioned however that this argument is made in a very informal manner. A formal argument must be based on a formal computational model of real-valued continuous functions. In this paper, we pursue the question of the existence of continuous one-way functions in this direction.

Our model of computation for real functions is based on the oracle Turing machines and is a generalization of the model used in recursive analysis. In this model, a real number $x \in [0, 1]$ is *polynomial-time computable* if there exists a Turing machine M which computes an approximate value d to x, with error $\le 2^{-n}$, in time $p(n)$ for some polynomial function p. A real function $f : [0, 1] \to R$ is *polynomial-time computable* if there exists an oracle Turing machine M which computes an approximate value e to $f(x)$, with error $\le 2^{-n}$, in time $p(n)$ for some polynomial function p, when an approximate value d to x with error $\le 2^{-p(n)}$ is given to M by the oracle. A polynomial complexity theory of real functions based on this model of computation has been developed by Ko and Friedman [5]. In this theory, the computational complexity of many basic numerical operations is characterized by the discrete complexity classes such as *P*, *NP* and *PSPACE*. (See Section 3 for the formal definitions; and see Ko [4] for a detailed survey of this theory).

Following the direction of this complexity theory, we study the necessary and sufficient conditions for the existence of continuous one-way functions. Before we begin to describe our main results, it is necessary to establish some technical definitions. One of the most basic properties of a polynomial-time computable function f on $[0, 1]$ is that f must have a polynomial modulus of continuity: if $|x - y| \le 2^{-p(n)}$ then $|f(x) - f(y)| \le 2^{-n}$, where p is a fixed polynomial function. Thus, in order to exclude trivial one-way functions on $[0, 1]$, we require that a one-way function f on $[0, 1]$ must have a polynomial *inverse modulus of continuity*: the inverse function f^{-1} must have a polynomial modulus of continuity.

With this requirement, we can prove that there does not exist a one-way function from $[0, 1]$ to R, thus confirming the intuition discussed above. However, when we consider two-dimensional one-way functions, then the property of continuity does not help too much. More precisely, we prove the following:

(1) If $P = NP$ then there does not exist a one-to-one one-way function f from $[0, 1]^2$ to R^2.
(2) If $P = UP$ then there exists a one-to-one one-way function from $[0, 1]^2$ to R^2.
(3) If $P_1 \neq UP_1 \cap {}_{co-}UP_1$, then there exists a one-to-one one-way function from $[0, 1]^2$ to R^2 such that $f^{-1}(\langle 1, 1\rangle)$ is not a polynomial time compatible real number.

In the above condition $P_1 \neq UP_1 \cap {}_{co-}UP_1$, the subscription 1 indicates the complexity classes restricted to tally sets (sets over a singleton alphabet $\{0\}$). This condition $P_1 \neq UP_1 \cap {}_{co-}UP_1$ is equivalent to the existence of certain strong discrete one-way functions. See Section 2 for more discussions.

The above results seem to suggest that in the one-dimensional case, the reason that a one-way function does not exist is really due to the order-preserving property of the continuous function rather than the continuity property of the function. To further investigate this observation, we consider k-to-one continuous one-way functions. With a reasonable extension of the notion of polynomial inverse modulus of continuity (for the formal definition see Section 5), we define a k-to-one continuous one-way function to be a polynomial-time computable function f which has a polynomial inverse modulus of continuity and yet there exists some non-polynomial-time computable point x whose image $f(x)$ is polynomial-time computable. (Note that for $k > 1$, f does *not* have a continuous inverse function, so one-way functions are defined in such a way that a single inverse point is difficult to compute.)

Based on this definition, we show that for one-dimensional functions, if $k = 3$, then k-to-one functions still have a certain order-preserving property and hence they cannot be one-way functions. For $k = 4$, however, we can show a similar result as items (1) and (3):

(4) If $P = NP$ then, for all $k \ge 4$, there does not exist a k-to-one one-way function f from $[0, 1]$ to R.
(5) If $P_1 \neq UP_1 \cap {}_{co-}UP_1$, then there exists a four-to-one strong one-way function from $[0, 1]$ to R.

In addition, we consider two-dimensional k-to-one one-way functions. Recall that in the study of discrete one-way functions, it is often observed that the complexity of the inverse function is closely related to the complexity of the range of the function. In the case of one-dimensional functions, the range of a function on a compact interval must be a compact interval, and so there is no such relation. In the case of two-dimensional functions, we observe that the function f constructed above in result (3) does have an irregular range. In fact, the question of whether we can find a one-to-one two-dimensional one-way function whose range is exactly $[0, 1]^2$ is left open. Our last result shows that such a one-way function exists if we allow the function to be three-to-one.

(6) If $P_1 \neq UP_1 \cap {}_{co-}UP_1$, then there exists a three-to-one strong one-way function from $[0, 1]^2$ *onto* $[0, 1]^2$.

We review, in Section 2, the notions of one-way functions and the sufficient conditions for their existence in terms of relations on complexity classes. In Section 3 we present our formal model of computation for real numbers and real functions. The main results (1)-(6) are proved in Sections 4, 5 and 6.

2. Discrete one-way functions

In this section we review the definitions and basic characterizations of some discrete one-way functions. We assume that the reader is familiar with (deterministic and non-deterministic) Turing machines (TMs) and their complexity measures. We will use the alphabet $\Sigma = \{0, 1\}$. Let $s \in \Sigma^*$, we let $l(s)$ denote its length. (In the next three sections, we will write $|x|$ to denote the absolute value of a real number x, and so we use the nonstandard notation $l(s)$ for the length of a string.)

We first define some complexity classes which are useful in characterization of one-way functions. Let P and NP be the classes of sets accepted in polynomial time by deterministic and, respectively, nondeterministic TMs. The class NP has a simple characterization by polynomial-time predicates: a set A is in NP if and only if there exist a polynomial-time predicate R and a polynomial function p such that for all s, $s \in A \Leftrightarrow (\exists t, l(t) \le p(l(s)))R(s,t)$. For any $k \ge 1$, a nondeterministic M is called *k-unambiguous* if for any input s there exist at most k different accepting computations of M on s. Let ${}_{k-}UP$ be the class of sets accepted in polynomial-time by k-unambiguous nondeterministic TMs. Then, the class ${}_{k-}UP$ has a similar characterization: a set A is in ${}_{k-}UP$ if and only if there exist a polynomial-time predicate R and a polynomial function p such that for all s,

$$\begin{aligned} s \in A &\Leftrightarrow (\exists t, l(t) \le p(l(s)))\ R(s,t) \\ &\Leftrightarrow (\exists \text{ at most } k \text{ different } t, l(t) \le p(l(s)))R(s,t). \end{aligned}$$

When $k = 1$, the class ${}_{1-}UP$ is exactly the class UP of Valiant [6].

It is clear that $P \subseteq {}_{k-}UP \subseteq {}_{(k+1)-}UP$ for all $k \ge 1$. Whether these inclusions are proper is one of the major open questions in complexity theory. An interesting relation between these complexity classes is, however, known:

Proposition 2.1. *[7] For any $k > 1$, $P = UP$ if and only if $P = {}_{k-}UP$.*

Proof. The backward direction is immediate. The forward direction can be proved by induction. We describe the case $k = 2$ here. Assume that $P = UP$ and that $A \in {}_{2-}UP$ such that for some polynomial-time predicate Q and some polynomial function p, we have $s \in A \Leftrightarrow (\exists t, l(t) \le p(l(s)))Q(s,t) \Leftrightarrow (\exists \text{ at most 2 strings } t, l(t) \le p(l(s)))Q(s,t)$. Let $B = \{s \in A | (\exists \langle t_1, t_2\rangle, l(t_1) \le l(t_2) \le p(l(s)))\ [t_1 < t_2 \text{ and } Q(s,t_1) \text{ and } Q(s,t_2)]\}$ Then, B is in UP and hence by assumption $P = UP$, $B \in P$. Furthermore, let $C = A - B$. Then, $s \in C \Leftrightarrow (\exists t, l(t) \le p(l(s)))\ [Q(s,t) \text{ and } s \notin B]$. Since $B \in P$, the predicate $[Q(s,t) \text{ and } s \notin B]$ is polynomial-time computable. Thus, $C \in UP$ and again, by assumption $P = UP$, $C \in P$. Therefore, $A = B \cup C$ is in P. □

One-way functions are defined in Section 2. The existence of a one-way function is closely related to the complexity class UP.

Proposition 2.2. *[2,3] There exists a one-one one-way function if and only if $P \ne UP$.*

In general, one-way functions do not have to be one-to-one. We may define a k-to-one one-way function ϕ to be a k-to-one, polynomial-time computable, polynomially honest function which is not polynomial-time invertible. However, from a generalization of Proposition 2.2, we know that a k-to-one one-way function exists if and only if $P \ne {}_{k-}UP$, $k \ge 1$. So, by Proposition 2.1, a k-to-one one-way function exists if and only if a one-to-one one-way function exists (cf. Watanabe [7]).

The difficulty of inverting a one-way function seems, intuitively, partially due to the difficulty of computing the range of the one-way function [7]. Therefore, one might ask whether there exists a one-way function whose range is easily recognizable. The next proposition answers this question.

Proposition 2.3. *[2] The following are equivalent.*

(i) There exists a one-way function ϕ such that Range(ϕ) is polynomial-time computable.
(ii) $P \ne UP \cap {}_{co-}UP$.

The next question about one-way functions with even simpler forms of range is whether there exists a one-way function which is one-to-one and onto. From the above proposition, we easily see that such a one-way function does not exist if $P = UP \cap {}_{co-}UP$. However, it is unknown whether the converse holds.

For the purpose of constructing continuous one-way functions, we need the existence of some stronger types of one-way functions. One of them is the one-way function whose inverse on simple inputs 0^n is not polynomial-time computable. Let C be a complexity class. We write C_1 to denote the class of all *Tally* sets $A \subseteq 0^*$ in C.

Proposition 2.4. *The following are equivalent.*

(i) There exists a one-way function ϕ such that the function ϕ^{-1} restricted to $0^* \cap Range(\phi)$ is not polynomial-time computable.
(ii) $P_1 \neq UP_1$

Proof. (i) $\Rightarrow$ (ii) Assume that ϕ is a function satisfying condition (i) such that $q(l(\phi(s))) \geq l(s)$ for some polynomial function q.

Let $A = \{0^{\langle n,i,b\rangle} | i \leq q(n), b \in \{0,1\}, (\exists s, i \leq l(s) \leq q(n))[\phi(s) = 0^n$ and the ith bit of s is equal to $b]\}$. Then, apparently, $A \in UP_1$. We claim that $A \notin P_1$. To see this, we observe that $Range(\phi) \cap 0^*$ is polynomial-time recognizable using A as an oracle and that the function ψ defined on $Range(\phi) \cap 0^*$ such that $\psi(0^n) = \phi^{-1}(0^n)$ is computable in polynomial time using A as an oracle.

(ii) $\Rightarrow$ (i): Let $A \subseteq 0^*$ be in $UP_1 - P_1$. Then, there exist a polynomial-time predicate R and a polynomial p such that for all n,

$$\begin{aligned} 0^n \in A &\Leftrightarrow (\exists s, l(s) \leq p(n))\ R(0^n, s) \\ &\Leftrightarrow (\exists \text{ a unique } s, l(s) \leq p(n))\ R(0^n, s) \end{aligned} \tag{1}$$

Furthermore, with a simple padding technique, we may assume that the string s satisfying $R(0^n, s)$ is of length exactly $p(n)$.

Now define a function ϕ as follows: on input s, if $l(s) = p(n)$ and $R(0^n, s)$ for some n then $\phi(s) = 0^n$ else $\phi(s) = 1s$. Then, obviously, ϕ is a one-to-one, polynomial-time computable, polynomially honest function. Furthermore, $0^* \cap Range(\phi) = A$. We note that if ϕ is polynomial-time invertible on 0^* (i.e., if there exists a polynomial-time computable function ψ on 0^* such that $\phi(\psi(0^n)) = 0^n$ for all $0^n \in Range(\phi)$), then we can determine whether $0^n \in A$ easily by computing $s = \psi(0^n)$ and comparing $\phi(s)$ with 0^n. □

Proposition 2.5. *The following are equivalent.*

(i) There exists a one-way function ϕ such that $0^* \subseteq Range(\phi)$ and that the function ϕ^{-1} restricted to 0^* is not polynomial-time computable.
(ii) $P_1 \neq UP_1 \cap {}_{co-}UP_1$

Proof. The proof is similar to Proposition 2.3. In the direction (i) $\Rightarrow$ (ii), the main difference is that the set A is also in ${}_{co-}UP_1$ because for any triple $\langle n, i, b\rangle$, if $0^{\langle n,i,b\rangle} \notin A$ then there also exists a unique s such that $\phi(s) = 0^n$ and that either $l(s) < i$ or the ith bit of s is not equal to b.

For the direction (ii) $\Rightarrow$ (i), we observe that there exists another polynomial-time predicate Q such that

$$\begin{aligned} 0^n \notin A &\Leftrightarrow (\exists s, l(s) \leq p(n))\ Q(0^n, s) \\ &\Leftrightarrow (\exists \text{ a unique } s, l(s) \leq p(n))\ Q(0^n, s), \end{aligned}$$

because A is also in ${}_{co-}UP_1$. Now define the function ϕ as follows: on input s, if $l(s) = p(n)$ and $[R(0^n, s)$ or $Q(0^n, s)]$ for some n then $\phi(s) = 0^n$ else $\phi(s) = 1s$. Then, ϕ is again a one-to-one, polynomial-time computable, polynomially honest function, and $0^* \subseteq Range(\phi)$. Also, if ϕ is polynomial-time invertible on 0^*, then we can determine whether $0^n \in A$ by computing $s = \psi(0^n)$ and checking whether $R(0^n, s)$ or $Q(0^n, s)$. □

Finally we remark that the existence of strong one-way functions of the type defined above has some interesting applications in complexity theory. For instance, Allender and Watanabe [1] have studied carefully the question of whether there exists a polynomial-time computable, polynomially honest function $\phi : \Sigma^* \to 0^*$ such that ϕ is not polynomial-time invertible. They demonstrated several interesting relations between this question and other questions in complexity theory, including some concerning the generalized Kolmogorov complexity of strings.

3. Model of computation for continuous functions

In this section, we present the formal model of computation for real functions on $[0, 1]$ or $[0, 1]^2$. The computational complexity theory of real functions based on this model has been developed in Ko and Friedman [4,5]. Here we will only give a short review.

First we consider the representation of real numbers. Let D be the set of all dyadic rational numbers, i.e., $D = \{m/2^n | n, m \in N, n \geq 0\}$. A dyadic rational $d \in D$ is represented by a string of the form

$$\pm d_n \cdots d_1 d_0.e_1 \cdots e_m$$

with each d_i and each e_j in $\{0, 1\}$, and its value is

$$d = \pm \left(\sum_{i=0}^{n} d_i \cdot 2^i + \sum_{j=1}^{m} e_j \cdot 2^{-j} \right)$$

Each dyadic rational $d \in D$ has infinitely many representations. For each string s which represents some dyadic rational d we write $prec(s)$ to denote the precision of s, i.e., the number of bits to the right of the binary point. For convenience, we often speak of a dyadic rational d of precision n to denote one of its specific representation with $prec(s) = n$.

A real number x is represented by a *Cauchy function* $\phi : N \to D$ which has the property that for each n, $\phi(n)$ is a dyadic rational d of precision n such that $|d - x| \le 2^{-n}$. Such a function ϕ is said to *binary converge* to x. A real number x has an infinite number of Cauchy functions binary converging to it, and has a unique *standard Cauchy function* ϕ_x which is defined by $\phi_x(n)$ = the maximum dyadic rational d of precision n such that $d \le x$. A real number x is *computable* if there exists a computable function ϕ which binary converges to x. A real number x is *polynomial-time computable* if there exists a function ϕ which binary converges to x such that $\phi(n)$ is computable in time $p(n)$ for some polynomial p.

The computation of real functions is based on the model of oracle TMs. We first consider real functions defined on $[0, 1]$. A real function $f : [0, 1] \to R$ is *computable* if there exists an oracle TM M such that for any oracle ϕ which binary converges to some x in $[0, 1]$ and for any input n, M outputs a dyadic rational of precision n such that $|d - f(x)| \le 2^{-n}$. The function f is *polynomial-time computable* if this oracle machine always halts in $p(n)$ moves for some polynomial p, independent of the oracle function ϕ. One of the most important properties of computable functions is that they must be continuous. A polynomial-time computable function f must have a polynomial modulus of continuity: for some polynomial p, $|f(x) - f(y)| \le 2^{-n}$ whenever $x, y \in [0, 1]$ and $|x - y| \le 2^{-p(n)}$. This property can actually be used to characterize polynomial-time computable real functions.

Proposition 3.1. *[5] A real function $f : [0, 1] \to R$ is polynomial-time computable if and only if*

(i) f has a polynomial modulus of continuity and
(ii) there exist a polynomial-time Timing machine M and a polynomial p such that for any dyadic rational d of precision $p(n)$, $M(d)$ is a dyadic rational e of precision n such that $|e - f(d)| \le 2^{-n}$.

Two-dimensional polynomial-time computable functions are similarly defined. To avoid confusion, we will write $\langle x, y\rangle$ to denote a point in R^2 and reserve the notation (x, y) to denote the one-dimensional open interval $\{z | x < z < y\}$. A function $f : [0, 1]^2 \to R^2$ is *polynomial-time computable* if there exist a two-oracle machine M and a polynomial p such that for any oracles ϕ and ψ binary converging to x and y in $[0, 1]$, respectively, and for any input n, M outputs, in time $p(n)$, two dyadic nationals d and e of precision n and $|d - f_1(\langle x, y\rangle)| \le 2^{-n}$ and $|e - f_2(\langle x, y\rangle)| \le 2^{-n}$, where f_1 and f_2 are defined by $f(\langle x, y\rangle) = \langle f_1(\langle x, y\rangle), f_2(\langle x, y\rangle)\rangle$. (For convenience, we use the L_∞ norm for the distance in the two-dimensional space R^2.) Similarly, a function $f : S \to R^2$ has a polynomial modulus function q on $S \subseteq R^2$ if for any $\langle x_1, y_1\rangle, \langle x_2, y_2\rangle \in S$, $|f(\langle x_1, y_1\rangle) - f(\langle x_2, y_2\rangle)| \le 2^{-n}$ whenever $|\langle x_1, y_1\rangle - \langle x_2, y_2\rangle| \le 2^{-q(n)}$.

4. A two-dimensional one-way function

Following the notion of discrete one-way functions, we define continuous one-way functions as follows. Let S be either the interval $[0, 1]$ or the unit square $[0, 1]^2$, and let T be either the set R or R^2. A function $f : S \to T$ is a *weak one-way function* if f is a one-to-one, polynomial-time computable function such that f^{-1} has a polynomial modulus of continuity on $Range(f)$ but f^{-1} is not polynomial-time computable; f is a *strong one-way function* if, in addition, there exists a point y in $Range(f)$ such that y is polynomial-time computable but $f^{-1}(y)$ is not polynomial-time computable.

As we pointed out in Section 1, the requirement of f^{-1} having a polynomial modulus of continuity is necessary, like the concept of polynomial honesty in the discrete case, to exclude the possibility of trivial continuous one-way functions. In particular, for functions not having this property, Ko and Friedman [5] have proved the existence of trivial one-way functions.

Proposition 4.1. *[5] For any recursive function α, there exists a one-to-one, polynomial-time computable function f on* $[0, 1]$ *such that $f(0) < 0 < f(l)$ and $f^{-1}(0) >$ is a real number not computable in time $\alpha(n)$.*

On the other hand, if f does have this property then f^{-1} is easy to compute.

Theorem 4.1. *Let f be a one-to-one, polynomial-time computable function from* $[0, 1]$ *to R such that f^{-1} has a polynomial modulus of continuity on $f([0, 1])$. Then, the inverse function f^{-1} is also polynomial-time computable on $f([0, 1])$.*

Proof. Since f is one-to-one, it must be strictly increasing on $[0, 1]$ or strictly decreasing on $[0, 1]$. Without loss of generality, we assume that f is strictly increasing on $[0, 1]$.

The value $f^{-1}(y)$, for any y given by a function ψ such that $|\psi(m) - y| \le 2^{-m}$ for all $m > 0$, can be found by a binary search. Assume that f has a polynomial inverse modulus function q. Then, an iteration of the binary search may be described as follows. If l is the current lower bound for $f^{-1}(y)$ and r is the current upper bound for $f^{-1}(y)$, then we find an approximate value e to $f((l+r)/2)$ with error $\le 2^{-(q(n)+2)}$. The search halts with output $(l+r)/2$ if $|e - \psi(q(n)+2)| \le 2^{-(q(n)+2)}$ (because this implies that $|f((l+r)/2) - y| \le 2^{-q(n)}$ and hence $|(l+r)/2 - f^{-1}(y)| \le 2^{-n}$); and otherwise it continues by resetting l and r accordingly.

Note that the above procedure works uniformly for all y in polynomial time, and hence is a polynomial-time algorithm for computing f^{-1}. $\square$

The above proof uses a simple binary search algorithm which evaluates the function value $f(x_0)$ at some point x_0 and determine whether the inverse $f^{-1}(y)$ lies to the left or to the right of x_0. When one tries to extend this idea of binary search to two-dimensional functions, it becomes difficult to implement as the search procedure needs to evaluate the function f on a *line*, e.g., $x = x_0$, to determine whether the point $f^{-1}(\langle y_1, y_2\rangle)$ lies to the left or to the right of this line, and the evaluation of a function f on a line means, in general, the evaluation at an exponential number of points. In the following we formally tie this intuition with the difficulty to invert a discrete one-way function. First, we show that if $P = NP$ then (weak) two-dimensional one-way functions do not exist.

Theorem 4.2. *Let f be a one-to-one, polynomial-time computable function from $[0,1]^2$ to R^2 such that f^{-1} has a polynomial modulus of continuity. Then, f^{-1} is also polynomial-time computable if $P = NP$.*

Proof. Assume that f is computed by an oracle TM M in time p and that f^{-1} has a modulus function q, where both p and q are polynomial functions. For each dyadic rationals d_1, d_2, we write M^{d_1,d_2} to denote the computation of M using the standard Cauchy functions of d_1 and d_2 as oracle functions. We will compute an approximate point to $f^{-1}(\langle y_1, y_2\rangle)$ by nondeterministically guessing a point $\langle d_1, d_2\rangle$ and then checking that $f(\langle d_1, d_2\rangle)$ is close to $\langle y_1, y_2\rangle$; or, more formally, we will make a binary search for $\langle d_1, d_2\rangle$ by querying about the following prefix set:

$$\begin{aligned} A =&\{\langle 0^n, 0^m, d_1, d_2, e_1, e_2\rangle | (\exists d_1^*, d_2^*) prec(d_1^*) = prec(d_2^*) = p(q(n+1)),\\ &|M^{d_1^*, d_2^*}(q(n+1)) - \langle e_1, e_2\rangle| \le 2^{-q(n+1)}, |\langle d_1, d_2\rangle - \langle d_1^*, d_2^*\rangle| \le 2^{-m}\} \end{aligned}$$

We claim that a point $\langle d_1, d_2\rangle$ such that $|\langle d_1, d_2\rangle| - |\langle y_1, y_2\rangle| \le 2^{-n}$ can be found by making queries to A for at most $4n$ times. To see this, let e_1 and e_2 be dyadic rationals such that $|e_i - y_i| \le 2^{-q(n+1)}$ for $i = 1, 2$. If we have already obtained some $\langle d_1^{(k)}, d_2^{(k)}\rangle$ such that $|\langle d_1^{(k)}, d_2^{(k)}\rangle - \langle e_1, e_2\rangle| \le 2^{-k}$, then we make queries $\langle 0^n, 0^{k+1}, d_1, d_2, e_1, e_2\rangle \in ?A$ for each of the following pairs:

$$\begin{aligned} &\langle d_1^{(k)} - 2^{-(k+1)}, d_2^{(k)} - 2^{-(k+1)}\rangle, \langle d_1^{(k)} - 2^{-(k+1)}, d_2^{(k)} + 2^{-(k+1)}\rangle,\\ &\langle d_1^{(k)} + 2^{-(k+1)}, d_2^{(k)} - 2^{-(k+1)}\rangle, \langle d_1^{(k)} + 2^{-(k+1)}, d_2^{(k)} + 2^{-(k+1)}\rangle, \end{aligned} \tag{2}$$

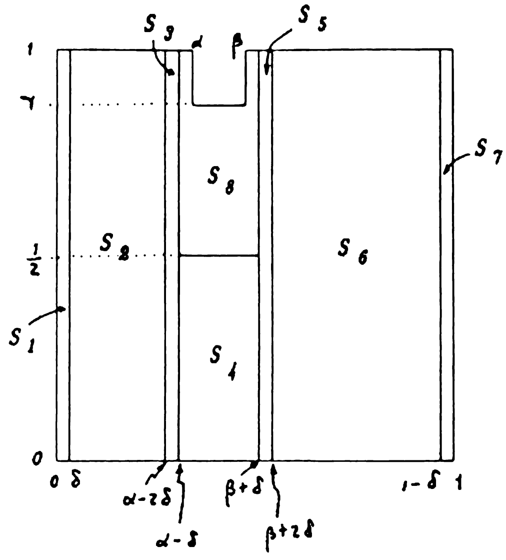
Then, at least one of the queries receives an affirmative answer and we let this pair be $\langle d_1^{(k+1)}, d_2^{(k+1)}\rangle$. It is easy to see that $A \in NP$. So, $P = NP$ implies that the above algorithm computes $f^{-1}(\langle y_1, y_2\rangle)$ in polynomial time. $\square$

Next we show that strong two-dimensional one-way functions exist if certain strong discrete one-way functions exist.

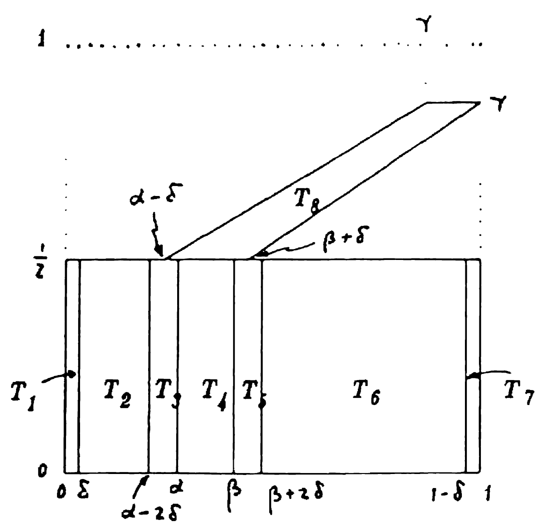
Theorem 4.3. *If $P_1 \ne UP_1 \cap {}_{co-}UP_1$ then there exists a one-to-one polynomial-time computable function f from $[0,1]^2$ to $[0,1]^2$ such that f^{-1} has a polynomial inverse modulus of continuity and that $f^{-1}(1,1)$ is unique and is not polynomial-time computable.*

Proof. The construction of the function f is quite involved. We first describe a basic construction of an infinite class of one-to-one functions $g = g(\alpha, \beta, \delta, \delta')$, where the parameters $\alpha, \beta, \delta, \delta'$ are dyadic rationals in the interval $[0,1]$ satisfying a $\alpha < \beta$ and $0 < 4\delta' < 4\delta \le \beta - \alpha < 1/2$.

Let $\gamma = 1 - (\beta - \alpha)$. Let S be the subset of the square $[0,1]^2$ with the square $[\alpha, \beta] \times [\gamma, 1]$ removed (but retaining the boundaries); more precisely, $S = [0,1]^2 - \{\langle x, y\rangle | \alpha < x < \beta, \gamma < y \le 1\}$. Also let T be the rectangle $[0,1] \times [0, 1/2]$ plus a trapezoid T' where T' is the trapezoid with the following four corners: $\langle \alpha - \delta, 1/2\rangle$, $\langle \beta + \delta, 1/2\rangle$, $\langle 1, \gamma\rangle$, $\langle \gamma, \gamma\rangle$. The domain of the function g is S and its range is T. We divide sets S and T each into 8 regions: $S_1, \cdots, S_8$, and $T_1, \cdots, T_8$. The function g will map each region S_i, $1 \le i \le 8$, into the region T_i. The regions $S_1, \cdots, S_7$, and $T_1, \cdots, T_7$ are rectangles:



(a) Domain of function g



(b) Range of function g

Fig. 1. Function g.

$$
\begin{aligned}
S_1 &= [0,\delta]\times[0,1], & T_1 &= [0,\delta]\times[0,1/2],\\
S_2 &= [\delta,\alpha-2\delta]\times[0,1], & T_2 &= [\delta,\alpha-2\delta]\times[0,1/2],\\
S_3 &= [\alpha-2\delta,\alpha-\delta]\times[0,1], & T_3 &= [\alpha-2\delta,\alpha]\times[0,1/2],\\
S_4 &= [\alpha-\delta,\beta+\delta]\times[0,1/2], & T_4 &= [\alpha,\beta]\times[0,1/2],\\
S_5 &= [\beta+\delta,\beta+2\delta]\times[0,1], & T_5 &= [\beta,\beta+2\delta]\times[0,1/2],\\
S_6 &= [\beta+2\delta,1-\delta]\times[0,1], & T_6 &= [\beta+2\delta,1-\delta]\times[0,1/2],\\
S_7 &= [1-\delta,1]\times[0,1], & T_7 &= [1-\delta,1]\times[0,1/2].
\end{aligned}
$$

The region S_8 is the rectangle $[\alpha-\delta,\beta+\delta]\times[1/2,1]$ with the square $[\alpha,\beta]\times[\gamma,1]$ removed; i.e., $S_8=[\alpha-\delta,\beta+\delta]\times[1/2,1]-\{\langle x,y\rangle|\alpha<x<\beta,\gamma<y\le 1\}$. The region T_8 is just the trapezoid T'. The 8 regions of S and T are shown in Fig. 1.

In the following we will describe the function g as a combination of 8 functions g_i, $1\le i\le 8$, each defined on S_i. Each function g_i is in turn a combination of some linear functions. Here we say a function h maps a triangle A onto a triangle B is *linear* if h maps the three corners of A to three corners of B and maps all other points linearly dependent on the mapping on three corners. Similarly, a function h maps a trapezoid A onto a trapezoid B is *linear* if h maps the four corners of A to four corners of B in the same orientation such that the two parallel sides are mapped to two parallel sides and it maps all other points in A linearly dependent on the mapping of four corners. For a triangle T with three corners u,v,w, we denote it by $\triangle uvw$; for a trapezoid Z with the corners u,v,w,x, we denote it by $\square uvwx$. We describe each g_i as follows:

(1) To describe function g_1 we define $u_1=\langle 0,1\rangle, u_2=\langle\delta/2,1\rangle, u_3=\langle\delta,1\rangle, u_4=\langle\delta,0\rangle, u_5=\langle 0,0\rangle, v_1=\langle 0,0\rangle, v_2=\langle 0,1/2\rangle, v_3=\langle\delta,1/2\rangle, v_4=\langle\delta,0\rangle$ and $v_5=\langle\delta/2,0\rangle$. Then g_1 is a combination of three linear functions on three triangles: g_1 maps each u_i, $1\le i\le 5$, to v_i, and maps the triangle $S_{1,1}=\triangle u_1u_2u_5$ to the triangle $T_{1,1}=\triangle v_1v_2v_5$, the triangle $S_{1,2}=\triangle u_2u_3u_5$ to the triangle $T_{1,2}=\triangle v_2v_3v_5$ and the triangle $S_{1,3}=\triangle u_3u_4u_5$ to the triangle $T_{1,3}=\triangle v_3v_4v_5$

(2) The function g_2 is linear on S_2 with the four corners mapped as follows: $g_2(\langle\delta,0\rangle)=\langle\delta,0\rangle, g_2(\langle\alpha-2\delta,0\rangle)=\langle\alpha-2\delta,0\rangle, g_2(\langle\alpha-2\delta,1\rangle)=\langle\alpha-2\delta,1/2\rangle$ and $g_2(\langle\delta,1\rangle)=\langle\delta,1/2\rangle$. That is, for any $\langle x,y\rangle$ in S_2, $g_2(\langle x,y\rangle)=\langle x,y/2\rangle$.

(3) The function g_3 is a combination of three linear functions. Let $u_1=\langle\alpha-2\delta,1\rangle, u_2=\langle\alpha-\delta,1\rangle, u_3=\langle\alpha-\delta,1/2\rangle, u_4=\langle\alpha-\delta,0\rangle, u_5=\langle\alpha-2\delta,0\rangle, v_1=\langle\alpha-2\delta,1/2\rangle, v_2=\langle\alpha-\delta,1/2\rangle, v_3=\langle\alpha,1/2\rangle, v_4=\langle\alpha,0\rangle$ and $v_5=\langle\alpha-2\delta,0\rangle$. The function g_3 maps each u_i, $1\le i\le 5$, to v_i and is a linear function from triangle $S_{3,1}=\triangle u_1u_2u_5$ to triangle $T_{3,1}=\triangle v_1v_2v_5$, from triangle $S_{3,2}=\triangle u_2u_3u_5$ to triangle $T_{3,2}=\triangle v_2v_3v_5$, and from triangle $S_{3,3}=\triangle u_3u_4u_5$ to triangle $T_{3,3}=\triangle v_3v_4v_5$.

(4) The function g_4 is linear on S_4 with the four corners mapped as follows: $g_4(\langle\alpha-\delta,0\rangle)=\langle\alpha,0\rangle, g_4(\langle\beta+\delta,0\rangle)=\langle\beta,0\rangle, g_4(\langle\beta+\delta,1/2\rangle)=\langle\beta,1/2\rangle$ and $g_4(\langle\alpha-\delta,1/2\rangle)=\langle\alpha,1/2\rangle$.

(5) The function g_5 is, similar to g_3, a combination of three linear functions. Let $u_1=\langle\beta+\delta,0\rangle, u_2=\langle\beta+\delta,1/2\rangle, u_3=\langle\beta+\delta,1\rangle, u_4=\langle\beta+2\delta,1\rangle, u_5=\langle\beta+2\delta,0\rangle, v_1=\langle\beta,0\rangle, v_2=\langle\beta,1/2\rangle, v_3=\langle\beta+\delta,1/2\rangle, v_4=\langle\beta+2\delta,1/2\rangle$ and $v_5=\langle\beta+2\delta,0\rangle$. The function g_5 maps each $u_i, 1\le i\le 5$, to v_i and is a linear function from triangle $S_{5,1}=\triangle u_1u_2u_5$ to

triangle $T_{5,1} = \triangle v_1 v_2 v_5$, from triangle $S_{5,2} = \triangle u_2 u_3 u_5$ to triangle $T_{5,2} = \triangle v_2 v_3 v_5$, and from triangle $S_{5,3} = \triangle u_3 u_4 u_5$ to triangle $T_{5,3} = \triangle v_3 v_4 v_5$.

(6) The function g_6 is the same as g_2; namely, $g_6(\langle x, y\rangle) = \langle x, y/2\rangle$.

(7) The function g_7 is Similar to g_1. Let $u_1 = \langle 1, 0\rangle, u_2 = \langle 1-\delta/2, 0\rangle, u_3 = \langle 1-\delta, 0\rangle, u_4 = \langle 1-\delta, 1\rangle, u_5 = \langle 1, 1\rangle, v_1 = \langle 1, 1/2\rangle, v_2 = \langle 1, 0\rangle, v_3 = \langle 1-\delta, 0\rangle, v_4 = \langle 1-\delta, 1/2\rangle$ and $v_5 = \langle 1-\delta/2, 1/2\rangle$. The function g_7 maps each u_i, $1 \le i \le 5$, to v_i and is a linear function from triangle $S_{7,1} = \triangle u_1 u_2 u_5$ to $T_{7,1} = \triangle v_1 v_2 v_5$, from triangle $S_{7,2} = \triangle u_2 u_3 u_5$ to $T_{7,2} = \triangle v_2 v_3 v_5$, and from triangle $S_{7,3} = \triangle u_3 u_4 u_5$ to $T_{7,3} = \triangle v_3 v_4 v_5$.

(8) The function g_8 is a combination of three linear functions on trapezoids. Let $u_1 = \langle \alpha-\beta, 1\rangle, u_2 = \langle \alpha, 1\rangle, u_3 = \langle \alpha, \gamma\rangle, u_4 = \langle \beta, \gamma\rangle, u_5 = \langle \beta, 1\rangle, u_6 = \langle \beta+\delta, 1\rangle, u_7 = \langle \beta+\delta, 1/2\rangle, u_8 = \langle \alpha-\delta, 1/2\rangle, v_1 = \langle \alpha-\delta, 1/2\rangle, v_2 = \langle \gamma, \gamma\rangle, v_3 = \langle \gamma+\delta', \gamma\rangle, v_4 = \langle 1-\delta', \gamma\rangle, v_5 = \langle 1, \gamma\rangle, v_6 = \langle \beta+\delta, 1/2\rangle, v_7 = \langle \beta, 1/2\rangle$ and $v_8 = \langle \alpha, 1/2\rangle$. The function g_8 maps each u_i, $1 \le i \le 5$, to v_i and is a linear function from trapezoid $S_{8,1} = \Box u_1 u_2 u_3 u_8$ to trapezoid $T_{8,1} = \Box v_1 v_2 v_3 v_8$, from trapezoid $S_{8,2} = \Box u_3 u_4 u_7 u_8$ to trapezoid $T_{8,2} = \Box v_3 v_4 v_7 v_8$, from trapezoid $S_{8,3} = \Box u_4 u_5 u_6 u_7$ to trapezoid $T_{8,3} = \Box v_4 v_5 v_6 v_7$.

The above definitions of functions g_1, g_3, g_5, g_7 and g_8 are shown in Fig. 2.

From the definition of g_i's, we can easily check that function g is well-defined, in the sense that functions g_i's agree with each other on the boundaries of their regions S_i's. Since g is piecewise linear, it is continuous. Furthermore, both g and g^{-1} have the modulus of continuity $m(n) = n+1+log(1/\delta)$: we observe that, within any region, the maximum distance between two points $g(\langle x_1, y_1\rangle)$ and $g(\langle x_2, y_2\rangle)$ is ≤ 2, if the distance between $\langle x_1, y_1\rangle$ and $\langle x_2, y_2\rangle$ is $\le \delta$ (this happens in g_8 Which maps the points u_1 and u_2 of distance δ to v_1 and v_2 of distance $\le 3/2$), and the minimum distance between two points $g(\langle x_1, y_1\rangle)$ and $g(\langle x_2, y_2\rangle)$ is $\ge \delta$, if the distance between $\langle x_1, y_1\rangle$ and $\langle x_2, y_2\rangle$ is ≥ 1. So the modulus function m is correct for both g and g^{-1} because g is piecewise linear on each region $S_i, 1 \le i \le 8$.

From this basic function g, we are ready to describe the function f. First we recall that from the assumption $P_1 \ne UP \cap {}_{co-}UP_1$, there exists a one-to-one, polynomial-time computable function $\phi : \Sigma^* \to \Sigma^*$ such that $0^* \subseteq Range(\phi)$ and that the function $\phi^{-1}(0^*)$ is not polynomial-time computable. Furthermore, without loss of generality, we may assume that there exists a polynomial-time function p such that $l(\phi^{-1}(0^n)) = p(n)$ and $\phi^{-1}(0^n) \ne 0^{p(n)}$ and $\phi^{-1}(0^n) \ne 1^{p(n)}$. Define a function $\psi : N \to N$ by $\psi(n) =$ the integer m whose $p(n)$-bit binary representation, with leading zeros, is equal to $\phi^{-1}(0^n)$.

Now we define the following function $r(n)$ and sequences a_n, b_n and c_n:

$$r(1) = 0; \quad r(n) = \sum_{i=1}^{n-1} p(i), n > 0;$$

$$c_n = 1 - 2^{-r(n)};$$

$$a_1 = 0; \quad a_{n+1} = a_n + \psi(n) \cdot 2^{-r(n+1)};$$

$$b_n = a_n + 2^{-r(n)}.$$

Then, let W_n be the square $[a_n, b_n] \times [c_n, 1]$ and V_n be the square $[c_n, 1] \times [c_n, 1]$. Let U_n be the set $W_n - W_{n+1}$ plus its boundary. Let g_n be the function g defined above with the parameters $\alpha_n = \psi(n) \cdot 2^{-p(n)}, \beta_n = (\psi(n)+1) \cdot 2^{-p(n)}, \delta'_n = 2^{-(p(n)+2)}$ and $\delta'_n = 2^{-(p(n)+p(n+1)+3)}$; that is, $g_n = g(\alpha_n, \beta_n, \delta_n, \delta'_n)$. Let $\gamma = 1 - (\beta_n - \alpha_n)$. We define a function f_n which maps U_n to a subset T_n of V_n by

$$f_n(\langle x, y\rangle) = \langle 1 - 2^{-r(n)}(1-u_n), 1 - 2^{-r(n)}(1-v_n)\rangle, \tag{*}$$

where

$$\langle u_n, v_n\rangle = g_n(\langle 2^{r(n)}(x-a_n), 2^{r(n)}(y-c_n)\rangle).$$

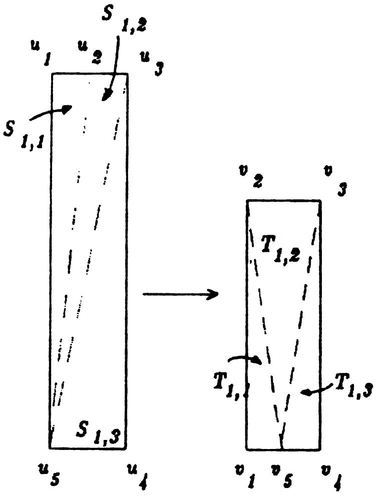
That is, the function f_n from U_n to V_n, is a linear transformation of g_n from $[0,1]^2$ to $[0,1]^2$. The function f is the combination of f_n on $U_n, n \ge 1$, plus $f(\langle x_0, 1\rangle) = \langle 1, 1\rangle$, where $x_0 = \lim_{n\to\infty} a_n$. Note that $[0,1]^2 = \cup_{n=1}^{\infty} U_n \cup \{\langle x_0, 1\rangle\}$, and so f is defined on every point in $[0,1]^2$. Furthermore, we claim that the function f is defined in such a way that the values of $f_n(\langle x, y\rangle)$ and $f_{n+1}(\langle x, y\rangle)$ agree on the boundary between S_n and S_{n+1} and therefore f is well-defined.

Proof. This can be verified by inspection about the functions f_n and f_{n+1} on the following points: $\langle a_{n+1}, 1\rangle, \langle a_{n+1}, c_{n+1}\rangle, \langle b_{n+1}, c_{n+1}\rangle$ and $\langle b_{n+1}, 1\rangle$. For instance, consider the point $\langle x, y\rangle = \langle a_{n+1}, c_{n+1}\rangle$. We verify that

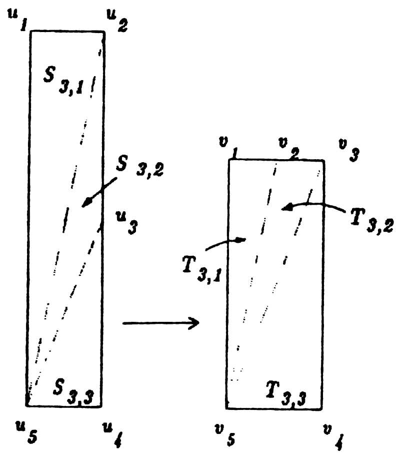
$$\begin{aligned}\langle u_n, v_n\rangle &= g_n(\langle 2^{r(n)}(a_{n+1}-a_n), 2^{r(n)}(C_{n+1}-c_n)\rangle) \\ &= g_n(\langle \psi(n) \cdot 2^{-p(n)}, 1 - 2^{p(n)}\rangle) = g_n(\langle \alpha_n, \gamma_n\rangle) \\ &= \langle \gamma_n + \delta'_n, \gamma_n\rangle), \quad \text{(this is the point } v_3 \text{ in the definition of } g_8)\end{aligned}$$

and hence

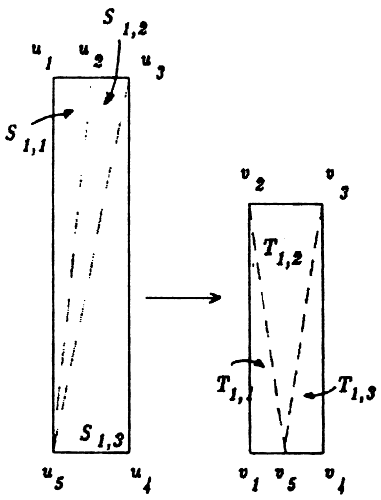
$$\begin{aligned}f_n(\langle a_{n+1}, c_{n+1}\rangle) &= \langle 1 - 2^{-r(n)}(1 - \gamma_n - \delta'_n, 1 - 2^{-r(n)}(1-\gamma_n)\rangle \\ &= \langle 1 - 2^{-r(n+1)} + 2^{-(r(n+1)+p(n+1)+3)}, 1 - 2^{-r(n+1)}\rangle \\ &= \langle 1 - 2^{-r(n+1)} + 2^{-(r(n+2)+3)}, 1 - 2^{-r(n+1)}\rangle;\end{aligned}$$



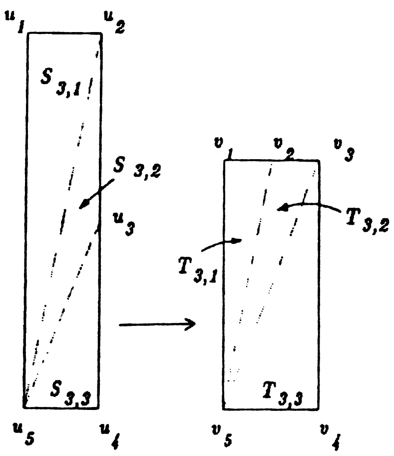
(a) Function g_1



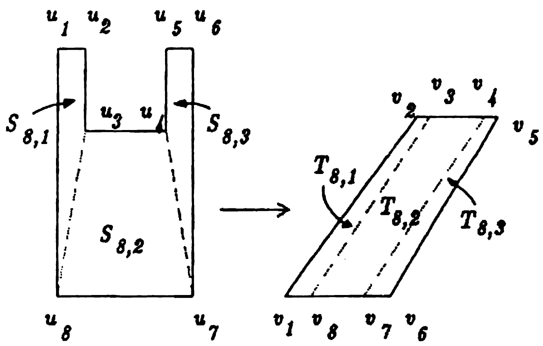
(b) Function g_3



(c) Function g_5



(d) Function g_7



(e) Function g_8

Fig. 2. Definitions of functions.

and

$$\begin{aligned}\langle u_{n+1}, v_{n+1}\rangle &= g_{n+1}(\langle 2^{r(n+1)}(a_{n+1}-a_{n+1}), 2^{r(n+1)}(c_{n+1}-c_{n+1})\rangle)\\ &= g_{n+1}(\langle 0,0\rangle) = \langle \delta_{n+1}/2, 0\rangle = \langle 2^{-(p(n+1)+3)}, 0\rangle,\end{aligned}$$

and hence

$$\begin{aligned}f_{n+1}(\langle a_{n+1}, c_{n+1}\rangle) &= \langle 1-2^{-r(n+1)}(1-2^{-(p(n+1)+3)}), 1-2^{-r(n+1)}(1-0)\rangle)\\ &= \langle 1-2^{-r(n+1)}+2^{-(r(n+2)+3)}), 1-2^{-r(n+1)}\rangle)\\ &= f_n(\langle a_{n+1}, c_{n+1}\rangle).\end{aligned}$$

The values at other points can be similarly verified. □

Next we claim that both f and f^{-1} have polynomial moduli of continuity. This can be seen from the moduli of continuity of g and g^{-1}. Namely, for any two points $\langle x_1, y_1\rangle$ and $\langle x_2, y_2\rangle$ in U_n, if $|\langle x_1, y_1\rangle - \langle x_2, y_2\rangle| \le 2^{-(m+1+\log(1/\delta_n))} \cdot 2^{-r(n)} = 2^{-(m+r(n+1)+3)}$, then $|f(\langle x_1, y_1\rangle - \langle x_2, y_2\rangle)| \le 2^{-(m+r(n))}$; and if $|f(\langle x_1, y_1\rangle - \langle x_2, y_2\rangle)| \ge 2^{-(m+r(n))}$ then $|f(\langle x_1, y_1\rangle - \langle x_2, y_2\rangle)| \ge 2^{-(m+r(n+1)+3)}$, for all $m \ge 0$ (note that if both $\langle x_1, y_1\rangle$ and $\langle x_2, y_2\rangle$ are in U_n, then their distance and the distance between their image points are bounded by $2^{-r(n)}$). So the claim is proven.

Now we want to show that f is polynomial-time computable on $[0,1]^2$. We first claim that for any given dyadic national point $\langle d_1, d_2\rangle$ and any integer m, we can either find an integer $n < m$ such that $\langle d_1, d_2\rangle \in U_n - U_{n+1}$ and obtain the values of a_i and b_i for $1 \le i \le n$, or conclude that $\langle d_1, d_2\rangle \in U_m$ and obtain the values of a_i and b_i for $1 \le i \le m$, and we can do it in time $q_1(m)$ for some polynomial q_1.

Proof. This can be done recursively. Assume that we already know that $\langle d_1, d_2\rangle \in U_n$ and know the values of a_n and b_n. To determine whether $\langle d_1, d_2\rangle \in U_{n+1}$, we simply find the maximum integer k such that $a_n + k \cdot 2^{-r(n+1)} \le d_1$ and let s_k and s_{k-1} be the $p(n)$-bit binary representations of the integer k and $k-1$ (with leading zeros), respectively, and compote $\phi(s_k)$ and $\phi(s_{k-1})$. If $\phi(s_k) = 0^n$ then $a_{n+1} \le d_1 \le b_{n+1}$, else if $\phi(s_{k-1}) = 0^n$ and $d_1 = a_n + k \cdot 2^{-r(n+1)}$ then $d_1 = b_{n+1}$, and else $\langle d_1, d_2\rangle \in U_n - U_{n+1}$. If $a_{n+1} \le d_1 \le b_{n+1}$ then we conclude that $\langle d_1, d_2\rangle \in U_{n+1}$ if and only if $c_{n+1} \le d_2 \le 1$. Note that c_{n+1} is computable in polynomial time from 0^n and a_{n+1} and b_{n+1} have been found in polynomial time. Therefore, we can recursively apply this procedure to determine the integer $n < m$ such that $\langle d_1, d_2\rangle \in U_n - U_{n+1}$, or to determine that $\langle d_1, d_2\rangle \in U_m$. □

Once we have established that $\langle d_1, d_2\rangle \in U_n - U_{n+1}$ and knowing the values of a_n and b_n we can compute $f(\langle d_1, d_2\rangle) = f_n(\langle d_1, d_2\rangle)$ by the formula $(*)$. That is, we only need to show that $g_n(\langle d_1', d_2'\rangle)$ can be computed correct to within an error 2^{-m} in $q_2(n+m)$ moves for some polynomial q_2, where $d_1' = 2^{r(n)}(d_1 - a_n)$ and $d_2' = 2^{r(n)}(d_2 - c_n)$. We observe that the proof of the earlier claim actually gives a procedure to determine whether $a_{n+1} \le d_1 \le b_{n+1}$. Use this procedure, we can actually determine which of the following cases holds.

Case 1. $\langle d_1', d_2'\rangle \in S_1$ or $\langle d_1', d_2'\rangle \in S_7$.

In this case, we calculate $g_n(\langle d_1', d_2'\rangle)$ in polynomial-time without knowing what a_{n+1} is.

Case 2. $\langle d_1', d_2'\rangle \in S_2 \cup S_6$.

In this case, we compute $g_n(\langle d_1', d_2'\rangle) = \langle d_1', d_2'/2\rangle$. (Note that we do not need to know whether $\langle d_1', d_2'\rangle \in S_2$ or $\langle d_1', d_2'\rangle \in S_6$. As long as $|d_1 - a_n| > \varepsilon_n$, $|d_1 - b_n| > \varepsilon_n$, $d_l - 2\varepsilon_n$ and $d_l + 2\varepsilon_n$ are not in (a_{n+1}, b_{n+1}), we decide that Case 2 holds.)

Case 3. $\langle d_1', d_2'\rangle \in S_3 \cup S_4 \cup S_5 \cup S_8$.

In this case, we must have already known the values a_{n+1} and $\psi(n)$, and hence knew whether $\langle d_1', d_2'\rangle$ is in S_3 or in S_4 or in S_5 or in S_8. Thus, we can calculate $g_n(\langle d_1', d_2'\rangle)$ from the definition accordingly.

Finally, since f has a polynomial modulus of continuity, the above calculation $f(\langle d_1, d_2\rangle)$ can be used to approximate the value $f(\langle x, y\rangle)$ at any point $\langle x, y\rangle$. This completes the proof that f is polynomial-time computable.

The only thing left to show is that $f^1(\langle 1,1\rangle)$ is not polynomial-time computable. To see this, we note that if $f(\langle x, y\rangle) = \langle 1,1\rangle$ then $x = \lim a_n$ and $y = \lim c_n$. Therefore, $y = 1$. Assume otherwise that x is a polynomial-time computable real number. Then, we can compute, for any given n, a dyadic rational d of precision $r(n+1)$ such that $|d - x| \le 2^{-r(n+1)}$, in $q_3(n)$ moves for some polynomial q_3. Now, take the maximum dyadic rational e of precision $r(n)$ such that $e \le d$ then e must be exactly $a(n)$ because $a_n + 2^{-r(n+1)} \le a_{n+1} < x < a_{n+1} + 2^{-r(n+1)} < b_n$ and $|d - x| \le 2^{-r(n+1)}$ imply that $a_n \le d \le b_n$. In other words, we can compute a_n in $q_4(n)$ moves for some polynomial q_4. From a_n and a_{n+1}, we can calculate the value $\psi(n)$ easily. This shows that the function $\psi^{-1}(0^n)$ is polynomial time computable and is a contradiction. This completes the proof of the theorem. □

From the above proof, we were not able to perform a binary search to find the inverse value $f^1(\langle 1,1\rangle)$. This is partly due to the fact that a binary search in the two-dimensional space requires the evaluation of f at potentially an exponential number of points, and also partly due to the fact that $Range(f)$ does not include a neighborhood $\{\langle y_1, y_2\rangle \mid |y_j - 1| \le \varepsilon, j = 1,2\}$ of $\langle 1,1\rangle$. It seems that either reason is strong enough to guarantee a two-dimensional one-way function. Nevertheless, we are not able to construct a one-way function which is one-to-one from $[0,1]^2$ onto $[0,1]^2$. (It is interesting to compare this inability of finding continuous one-way functions with the domain and range both equal to $[0,1]^2$ with the ease of discrete one-to-one onto one-way functions discussed in Section 2.)

Question Does there exist a two-dimensional one-way function whose range is exactly $[0,1]^2$?

We will give a partial answer to this question in Section 6.

5. Many-to-one one-way functions

In this and the next sections, we consider continuous one-way functions which are not necessarily one-to-one. In the case of discrete functions, we have seen in Section 2 that a k-to-one one-way function exists if and only if a one-to-one one-way function exists. In the case of continuous functions, we will see quite different results. In particular, we will show, under the assumption $P_1 \ne UP_1 \cap_{co-} UP_1$, the existence of one-dimensional four-to-one one-way functions (in contrast to Theorem 4.1) and, in Section 6, the existence of two-dimensional three-to-one one-way functions (in contrast to the above Question). Intuitively, the one-to-oneness of a continuous function implies strong regularity between the function values at neighboring points while the k-to-oneness, for $k > 1$, allows more irregularity.

Before we can prove our main results, we must define what a one-way function f is if f is allowed to be many-to-one. In this section, we only consider one-dimensional functions. First we need to introduce a new concept of polynomial inverse modulus of continuity which is an extension of the requirement that the inverse of a one-way function must have a polynomial modulus of continuity. The purpose of this requirement is to exclude trivial one-way functions. Intuitively, a function f has an inverse modulus of continuity q if for any points $x, x' \in [0,1]$ and for any $n > 0$, $|x - x'| > 2^{-n}$ implies $|f(x) - f(x')| > 2^{-q(n)}$; that is, the graph of f does not occur to the right or the left of the square $S_x = \{\langle x', y'\rangle \mid |x - x'| \le 2^{-n}, |f(x) - y'| > 2^{-q(n)}\}$. However, for a k-to-one function f, this condition is too strong because we need allow f to have same value $f(x) = f(x')$ at some different points x and x'. To keep the spirit of this requirement while still allowing simple functions such as $f(x) = x^2$ on $[-1,1]$, we modify this requirement into the following two conditions.

(a) For any two points $x < x' \in [0,1]$ and for any $n > 0$, $|x - x'| > 2^{-n}$ implies that $|f(x) - f(z)| > 2^{-q(n)}$ for some $z \in (x, x']$.
(b) For any point $x \in [0,1]$, there exists an interval (a,b) such that $x \in (a,b)$ and for any $x' \in (a,b) \cap [0,1]$ and for any $n > 0$, $|x - x'| > 2^{-n}$ implies that $|f(x) - f(x')| > 2^{-q(n)}$.

Intuitively, the condition (a) is a global extension of the condition for one-to-one functions. Namely, condition (a) implies that if f is one-to-one on $[a,b]$ then f^{-1} on $f([a,b])$ has a modulus function q. The condition (b) is a local extension of the same condition; it allows f to have some point x' such that $f(x')$ is close to $f(x)$ either if x' is very close to x or if x' is outside a fixed neighborhood of x. In the following we will show that if f satisfies these two conditions with respect to a polynomial function q, then f^{-1} cannot be too difficult to compute (i.e., is polynomial-time computable if $P = NP$, and so it cannot be a “trivial” one-way function.

Definition 5.1. A function $f : [0,1] \to R$ is said to have a *polynomial inverse modulus of continuity* if there exists a polynomial function q such that f and q satisfy conditions (a) and (b) above.

For $k > 1$, since a k-to-one function f is not necessarily one-to-one, f^{-1} does not necessarily exist and so we only consider strong one-way functions.

Definition 5.2. Let $k > 0$. A k-to-one function $f : [0,1] \to R$ is a *strong one-way function* if it is polynomial-time computable, has a polynomial inverse modulus of continuity and for which there exists a non-polynomial-time computable point $x \in [0,1]$ such that $y = f(x)$ is polynomial-time computable.

Our first result shows that there is no such three-to-one one-way functions.

Theorem 5.1. *There does not exist a three-to-one one-way function.*

Proof. First, we prove that there is no two-to-one one-way function. Assume that f is two-to-one, is computable in polynomial time p, and has a polynomial inverse modulus function q. Also let y be a polynomial-time computable real number in $Range(f)$.

For each $x \in f^{-1}(\{y\})$, we find an interval $[a,b]$, with a and b both dyadic rationals, such that f and q satisfy condition (a) of Definition 5.1 on the interval $(a - \varepsilon, b + \varepsilon)$ for some $\varepsilon > 0$. It is clear then that x is the unique number in $[a,b]$ such that $f(x) = y$. Thus there are four cases regarding the relation between $f(x)$ and $f(z)$ for $z \in [a,b]$: (1) $f(z_1) < y < f(z_2)$ for all $z_1 \in [a,x)$ and all $z_2 \in [x,b)$, (2) $f(z_1) < y < f(z_2)$ for all $z_1 \in [a,x)$ and all $z_2 \in (x,b)$, (3) $f(z) < y$ for all $z \in [a,x) \cup (x,b]$, or (4) $f(z) > y$ for all $z \in [a,x) \cup (x,b)$. In the first two cases, we may use a binary search algorithm similar to the one in Theorem 4.1 to compute x and hence x is polynomial-time computable. So we may assume that, without loss of generality, $f(z) < y$ for all $z \in [a,x) \cup (x,b]$. Furthermore, we may assume that $f(a) \le f(b)$. Then, we can find another dyadic rational $a' \in [a,x)$ such that $f(a') = f(b)$. We note that by the two-to-oneness of f, f must be strictly increasing on $[a,x]$ and be strictly decreasing on $[x,b]$. The following algorithm performs a binary search for x, assuming that a' and b are explicitly given. □

Algorithm 1: A binary search for x.

Input: $n > n_0$.
/* need output α such that $|\alpha - x| \le 2^{-n}$ */
$l := a'$; $r := b$;
for $i := 1$ *to* n **do**
  $c := (l+r)/2$; $c_1 := c - 2^{-(n+1)}$; $c_2 := c + 2^{-(n+1)}$;
  compute e_1, e_2 such that $|f(c_i) - e_i| \ne 2^{-(q(n+2)+2)}$ for $i = 1, 2$;
  if $|e_1 - e_2| \ne 2^{-(q(n+2)+2)}$ **then**
    output c and halt
  else if $e_1 < e_2 - 2^{-(q(n+2)+2)}$ **then**
    let $l := c_1$
  else
    $e_1 > e_2 + 2^{-(q(n+2)+2)}$
  let $r := c_2$
Output: $(l+r)/2$ and halt

We note that if the above algorithm halts inside the loop and outputs c, then we have $|e_1 - e_2| \ne 2^{-(q(n+2)+2)}$ and hence $|f(c_1) - f(c_2)| \ne 2^{-q(n+2)}$, but $|c_1 - c_2| = 2^{-n}$. By condition (a) of Definition 5.1 and the monotonicity of f on $[a,x]$ and on $[x,b]$, c_1 and c_2 must locate on the two different sides of x. Since $|c_1 - c_2| = 2^{-n}$, we must have $|c - x| \le 2^{-(n+1)}$.

Assume that the algorithm halts outside the loop. We first show that we always have $l \le x \le r$ after each iteration of the loop. To see this, we claim that if before a particular iteration, we have $l \le x \le r$ and if we finish this iteration in Case 2 ($e_1 < e_2 - 2^{-(q(n+2)+2)}$) then we must have $c_1 \le x$.

Proof. Suppose otherwise that $x < c_1 < c_2$. Then, by the monotonicity of f on $[x,b]$ and condition (a) of Definition 5.1, we must have $f(c_1) > f(c_2) + 2^{-q(n+2)}$. However, $e_1 < e_2 - 2^{-(q(n+2)+2)}$ implies that $f(c_1) \le e_1 + 2^{-(q(n+2)+2)} < e_2 \le f(c_2) + 2^{-(q(n+2)+2)}$. Thus we have a contradiction. □

Similarly, if we finish the iteration in Case 3, we must have $x \le c_2$. Thus, the condition $l \le x \le r$ always holds. Next we observe that in each iteration, we reduce the size $r - l$ to its half plus $2^{-(n+1)}$. So, after n iterations, the size $r - l$ is at most $2^{-n} + 2^{-(n+1)} + 2^{-(n+2)} + \cdots + 2^{-2n} < 2^{-(n-1)}$. This shows that the output $(l+r)/2$ is within the distance 2^{-n} to x.

The above proved that the algorithm always outputs a correct approximate value for x. It is easy to verify that the algorithm always halts in polynomial time. So, we have proved the theorem for the case when f is two-to-one.

Now, consider the case for a three-to-one function f. Similarly to the case of two-to-one functions, we find, for each $x \in f^{-1}(y)$, an interval $[a,b]$ such that f and q satisfy conditions (*a*) and (*b*) of Definition 5.1. Again, without loss of generality, we may assume that $f(z) < x$ for all $z \in [a,x) \cup (x,b]$, and that $f(a) \le f(b)$. Let $a' = \max\{z \in [a,x) | f(z) = f(b)\}$. Then, we note that f must be strictly increasing on $[a',x]$ (otherwise the function f would be three-to-one on $[a',x)$, and so would be four-to-one on $[a',b]$, which is a contradiction). Similarly, f must be strictly decreasing on $[x,b]$. So, by the above proof for the case of two-to-one functions, we know that x is polynomial-time computable.

Remark. It is interesting to note that Algorithm 1 above actually did not use the value y to compute x; it only uses the fact that x is the local maximum point. In other words, for a three-to-one, polynomial-time computable function which has a polynomial inverse modulus of continuity, its local maximum points as well as its local maximum values are polynomial-time computable (cf. Ko [4]).

Next we show that if $P = NP$ then there is no k-to-one one-way functions, for any $k > 0$.

Theorem 5.2. *If* $P = NP$ *then there does not exist a* k*-to-one one-way function on* $[0,1]$ *for all* $k > 0$.

Proof. Let f be a k-to-one function from $[0,1]$ to R. Assume that f is computable by an oracle TM M in polynomial time p and has a polynomial inverse modulus function q. Let $y \in Range(f)$ be a polynomial-time computable point. For any x such that $f(x) = y$, let $[a,b]$ be an interval such that f on $(a - \varepsilon, b + \varepsilon)$ and q satisfy condition (b) of Definition 5.1 at x for some $\varepsilon > 0$.

We can compute x by nondeterministically guessing a dyadic point $d \in [a, b]$ of precision $prec(d) = p(q(n) + 2)$ and checking that $|M^d(q(n) + 2) - d_y| \le 2^{-(q(n)+2)}$, where $M^d(m)$ is the output of machine M on input n with the standard Cauchy function ϕ_d as the oracle function and d_y is a dyadic rational such that $|d_y - y| \le 2^{-(q(n)+2)}$. By the property of the inverse modulus function q, we know that this d must be within distance 2^{-n} of x. Therefore, if $P = NP$ then x is computable in polynomial time (cf. proof of Theorem 4.2). □

Remark. The above assumption $P = NP$ can actually be weakened into $P = UP$. Since the proof using the weaker assumption $P = UP$ is too technical, we only include a sketch here. Roughly speaking, we need to modify the above nondeterministic algorithm into the following form:

Let $i = q(n + 2)$ and $m = q(p(i) + 1)$ and assume that $|d_y - y| \le 2^{-(m+1)}$. Then, we need to guess some dyadic point $d \in [a, b]$ of precision $prec(d) = p(i)$ and check that

$$M^d(m + 1) \ge d_y - 2^{-i} \text{ and } M^{d'}(m + 1) \ge d_y - 2^{-i}, \tag{*}$$

where $d' = d + 2^{-p(i)}$. It can be proved that $f(d)$ is approximately $\ge d_y - 2^{-i}$ and $f(d')$ is approximately $< d_y - 2^{-i}$, and there are at most k such points d satisfying the above (*). Thus this computation can be done in polynomial time relative to an oracle in ${}_{k-}UP$. Since $P = UP$ implies $P = {}_{k-}UP$, we know that $P = UP$ implies this algorithm finds in polynomial time *a* point d which is close to x.

Finally we show that $P_1 \ne UP_1 \cap {}_{co-}UP_1$ is sufficient for the existence of *a* four-to-one one-way function on [0, 1].

Theorem 5.3. *Assume that $P_1 \ne UP_1 \cap {}_{co-}UP_1$. Then, there exists a four-to-one, polynomial-time computable function f from* [0, 1] *to R such that f has a polynomial inverse modulus of continuity and $f^{-1}(1)$ is unique and is not polynomial-time computable.*

Proof. The function f will be constructed as a piecewise linear function. Recall that from the assumption $P_1 \ne UP_1 \cap {}_{co-}UP_1$, there exists a one-to-one, polynomial-time computable function $\phi : \Sigma^* \to \Sigma^*$ such that $0^* \subseteq Range(\phi)$ and that the function ϕ^{-1} restricted to $0*$ is not polynomial-time computable. Furthermore, by a simple padding argument, we may assume that there exists a polynomial function p such that $|\phi^{-1}(0^n)| = p(n)$ and $\phi^{-1}(0^n) \ne 0^{p(n)}$ and $\phi^{-1}(0^n) \ne 1^{p(n)}$. We also assume that $p(n) > 2$ for all n. Define a function $\psi : N \to N$ by $\psi(n)$ = the integer m whose $p(n)$-bit binary representation, with leading zeros, is equal to $\phi^{-1}(0^n)$. Then $0 < \psi(n) < 2^{p(n)} - 1$.

Next we define the function $r(n)$ and sequences a_n, b_n, c_n and ε_n as follows.

$$\begin{aligned} &r(1) = 0; \quad r(n) = \Sigma_{i=1}^{n-1} p(i), n > 0; \\ &c_n = 1 - 2^{-r(n)}; \\ &\varepsilon_n = (1 - c_n)/4 = 2^{-(r(n)+2)}; \\ &a_1 = 0; \quad a_{n+1} = a_n + \psi(n) \cdot 2^{-r(n+1)}; \\ &b_n = a_n + 2^{-r(n)}. \end{aligned} \tag{3}$$

Then let f_n be the piecewise linear function on $[a_n, a_{n+1}] \cup [b_n, b_{n+1}]$ defined by the following six breakpoints

$$\begin{aligned} f_n(a_n) &= c_n, \\ f_n(a_{n+1} - \varepsilon_{n+1}) &= (a_{n+1} - a_n - \varepsilon_{n+1})/2 + c_n, \\ f_n(a_{n+1}) &= c_{n+1}, \\ f_n(b_{n+1}) &= (1 + c_{n+1})/2, \\ f_n(b_{n+1} + \varepsilon_{n+1}) &= (b_{n+1} + \varepsilon_{n+1} - a_n)/2 + c_n, \\ f_n(b_n) &= (1 + c_n)/2. \end{aligned} \tag{4}$$

The function f_n is shown in Fig. 3. (Note that $f(n)$ has slope 1/2 on $[a_n, a_{n+1} - \varepsilon_{n+1}] \cup [b_{n+1} + \varepsilon_{n+1}, b_n]$.) Let $f(x) = f_n(x)$ if $x \in [a_n, a_{n+1}] \cup [b_{n+1}, b_n]$, and $f(x_0) = 1$, where $x_0 = \lim_{n\to\infty} a_n$. We claim that f satisfies our requirements.

First, we check that f is well-defined, and is continuous on [0, 1]. This can be seen easily by verifying that $f_n(a_{n+1}) = c_{n+1} = f_{n+1}(a_{n+1})$, $f_n(b_{n+1}) = (1 + c_{n+1})/2 = f_{n+1}(b_{n+1})$, and that $\lim_{n\to\infty} c_n = 1$.

Next, we show that f is a four-to-one function. First, it is easy to see by inspection that each f_n is three-to-one from $[a_n, a_{n+1}] \cup [b_{n+1}, b_n]$ to $[c_n, (1 + c_{n+1})/2]$, and it is actually one-to-one *onto* the interval $(c_{n+1}, (1 + c_{n+1})/2]$ (in the sense that $f^{-1}(\{y\}) \cap ([a_n, a_{n+1}] \cup [b_{n+1}, b_n])$ has size ≤ 1 for all $y \in (c_{n+1}, (1+c_{n+1})/2]$). Furthermore, by the fact that $(1+c_n)/2 < c_{n+1}$ all n (because $p(n) > 2$ for all n and hence $r(n+1) > r(n)+2$), the function f_{n-1} is one-to-one to the interval $(c_n, c_{n+1}]$, and $[c_n, 1] \cap Range(f_i) = \emptyset$ for all $i < n - 1$. So, f is four-to-one to each interval $(c_n, c_{n+1}]$.

The next thing to check is that f is polynomial-time computable. We will show two properties of f: (a) the function f has a polynomial modulus of continuity, and (b) for any dyadic rational d of precision m, $f(d)$ is a dyadic rational of

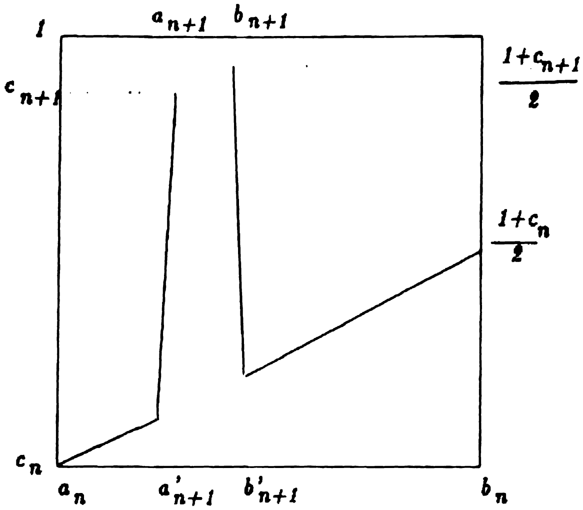


Fig. 3. Function of f_n of Theorem 4.3.

precision $\le q(m)$ and is computable in time $q(m)$ for some polynomial q. By Proposition 3.1, these two properties imply that f is polynomial-time computable.

Proof of part (a). We need only to check about the slopes of the function f_n. The slope of f_n is $1/2$ on $[a_n, a_{n+1} - \varepsilon_{n+1}]$ and on $[b_{n+1} + \varepsilon_{n+1}, b_n]$. The slope of f_n on $[a_{n+1} - \varepsilon_{n+1}, a_{n+1}]$ is $\le (1 - c_n)/\varepsilon_{n+1} = 2^{-r(n)} \cdot 2^{r(n+1)+2)} = 2^{p(n)+2}$. Similarly, the slope of f_n on $[b_{n+1}, b_{n+1} + \varepsilon_{n+1}]$ is $\ge -2^{p(n)+2)}$. Since f is piecewise linear, the function $p_1(n) = p(n) + 2$ is a modulus function for f.

Proof of part (b). This part is similar to the proof of Theorem 4.3. Note that we have defined the sequences a_n, b_n exactly the same as in Theorem 4.3. And, in that proof, we showed that, given a dyadic rational d, we can find in polynomial time the integer n such that $d \in [a_n, b_n] - (a_{n+1}, b_{n+1})$ and can determine the values of a_i, b_i for $i \le n$. Similarly, we can determine whether $d \in [a_{n+1} - \varepsilon_{n+1}, a_{n+1}]$ or $d \in [b_{n+1}, b_{n+1} + \varepsilon_{n+1}]$. If this is the case, then compute $f_n(a_{n+1} - \varepsilon_{n+1})$ and $f_n(a_{n+1})$ (or, $f_n(b_{n+1})$ and $f_n(b_{n+1} + \varepsilon_{n+1})$) according to the definition of f_n and linearly interpolate $f_n(d)$. If not, then we output $f(d) = f_n(d) = (d - a_n)/2 + c_n$. This completes the proof of part (b).

In the above, we have checked the slope of function f_n. In general, the absolute value of the slope of function f is $\ge 1/2$, and therefore, f has a polynomial inverse modulus of continuity.

Finally, to see that $f^{-1}(1)$ is unique and is not polynomial-time computable, we observe again that it is similar to the proof in Theorem 4.3. Note that $x_0 = \lim a_n$ is the unique number such that $f(x) = 1$, and that if x_0 is polynomial-time computable then we can, as shown in Theorem 4.3, compute sequence $\{a_n\}$ and hence the function $\psi(n)$ in polynomial time. $\square$

6. Two-dimensional onto one-way functions

Following the discussion of Section 5, we define two-dimensional k-to-one one-way functions as follows.

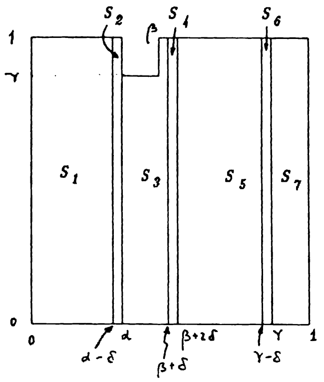
Definition 6.1. A function $f : [0, 1]^2 \to R^2$ is said to have a *polynomial inverse modulus of continuity* if there exists a polynomial function q such that f and q satisfy the following conditions:

(1) for any point $\langle x_1, x_2\rangle \in [0, 1]^2$ and for any $n > 0$, there exists a point $\langle z_1, z_2\rangle$ such that $|\langle z_1, z_2\rangle - \langle x_1, x_2\rangle| \le 2^{-n}$ and $|f(\langle z_1, z_2\rangle) - f(\langle x_1, x_2\rangle)| > 2^{-q(n)}$, and
(2) for any point $\langle x_1, x_2\rangle \in [0, 1]$, there exists $\delta > 0$ such that for any $\langle x'_1, x'_2\rangle \in [0, 1]^2$ such that $|\langle x_1, x_2\rangle - \langle x'_1, x'_2\rangle| < \delta$ and for any $n > 0$, $|\langle x_1, x_2\rangle - \langle x'_1, x'_2\rangle| > 2^{-n}$ implies that $|f(\langle x_1, x_2\rangle) - f(\langle x'_1, x'_2\rangle)| > 2^{-q(n)}$.

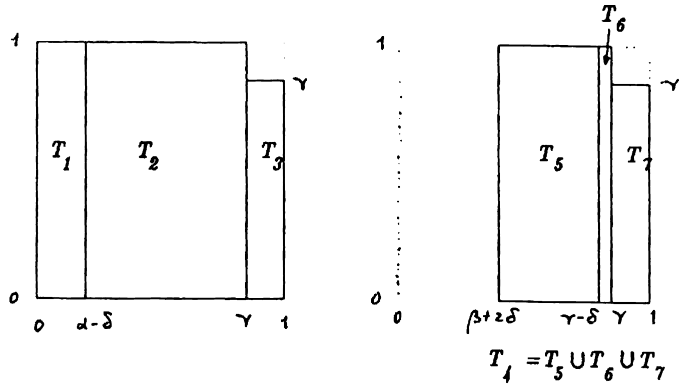
Definition 6.2. Let $k > 0$. A k-to-one function $f : [0, 1]^2 \to R^2$ is a *one-way function* if it is polynomial-time computable, has a polynomial inverse modulus of continuity and for which there exists a non-polynomial-time computable point $\langle x_1, x_2\rangle \in [0, 1]$ such that $f(\langle x_1, x_2\rangle)$ is polynomial-time computable.

Similarly to Theorems 4.2 and 5.2, k-to-one one-way functions on $[0, 1]^2$ do not exist if *P=NP*.

Theorem 6.1. *Let $k \ge 1$. If P=NP then there does not exist a k-to-one one-way function from $[0, 1]^2$ to R^2.*



(a) Domain of function g



(b) Range of function g

Fig. 4. Function g.

Proof. The proof is similar to those of Theorems 4.2 and 5.2. We omit it here. □

In the following we construct a three-to-one one-way function whose range is exactly $[0,1]^2$. This gives a partial answer to the Question at the end of Section 4.

Theorem 6.2. *If $P_1 \neq UP_1 \cap {}_{co-}UP_1$, then there exists a three-to-one polynomial-time computable function f from $[0,1]^2$ onto $[0,1]^2$ such that f has a polynomial inverse modulus of continuity and $f^{-1}(\langle 1,1\rangle)$ is unique and is not polynomial-time computable.*

Proof. The construction of the function f is similar to Theorem 4.3. We first describe a basic function g, whose role in the construction of f is similar to that of the basic function g in Theorem 4.3.

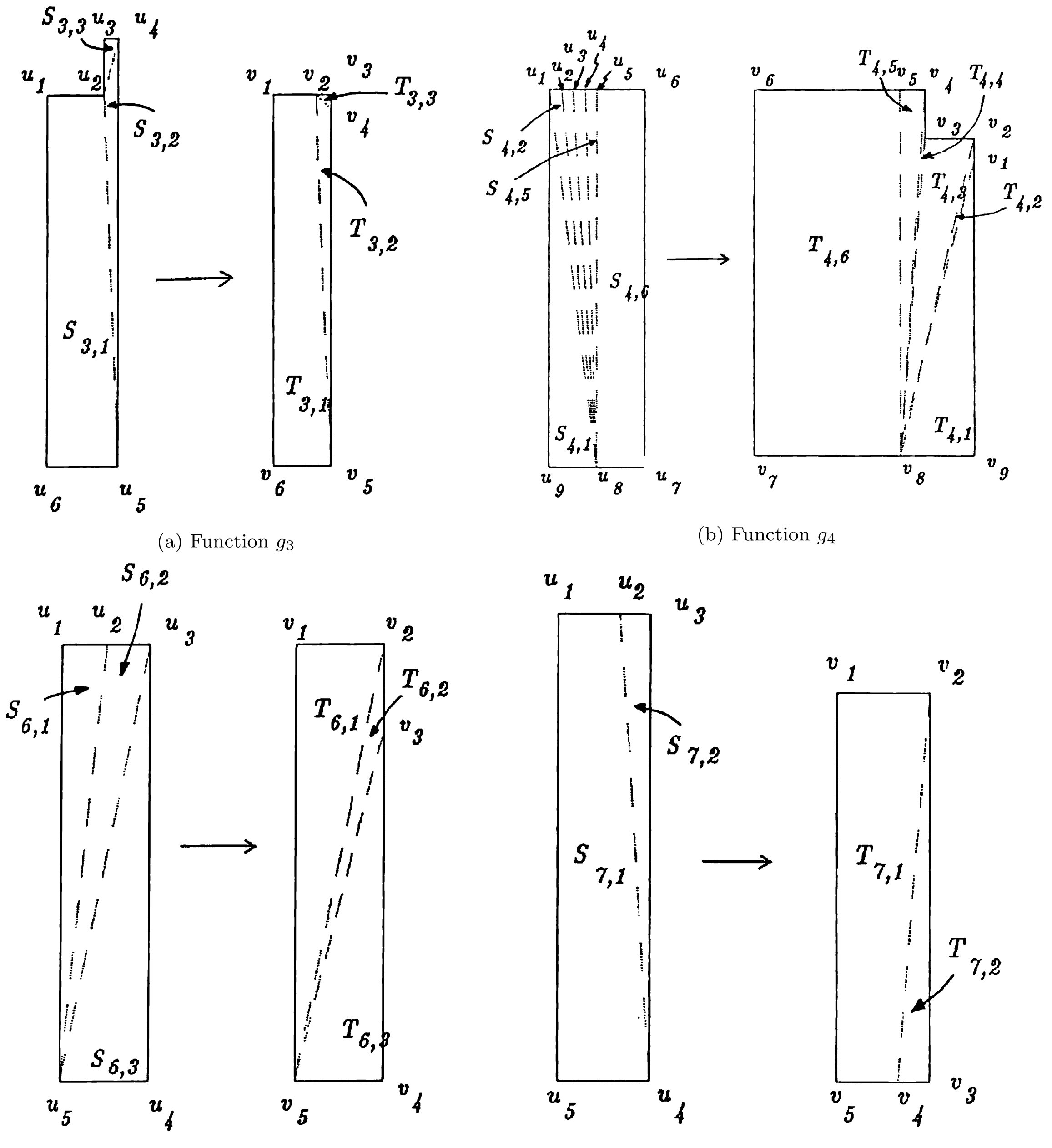
The function g is defined by four parameters; that is $g = g(\alpha,\beta,\delta,\delta')$, where the parameters $\alpha,\beta,\delta,\delta'$ are dyadic rationals in the interval $[0,1]$ satisfying $\alpha<\beta$ and $0<4\delta'<4\delta\le\beta-\alpha<1/2$.

Let $\gamma = 1-(\beta-\alpha)$. Let S be the subset of the square $[0,1]^2$ with the square $[\alpha,\beta]\times[\gamma,1]$ removed (but retaining the boundaries). Also let T be the square $[0,1]^2$ with the square $[\gamma,1]^2$ removed (but retaining the boundaries). $S = [0,1]^2 - \{\langle x,y\rangle | \alpha<x<\beta, \gamma<y\le 1\}$ and $T=[0,1]^2-\{\langle x,y\rangle|\gamma<x\le 1,\gamma<y\le 1\}$. The domain of the function g is S and its range is T. We divide set S into 7 regions $S_1,\cdots,S_7$, and define 7 subregions of $T: T_1,\cdots,T_7$. The function g will be a one-to-one mapping from each region S_i to T_i. Regions S_i for $1\le i\le 7$, $i\neq 3$, and regions T_j for $1\le j\le 7$, $j\neq 4$, are all rectangles; regions S_3 and T_4 are the unions of two rectangles. These regions are described as follows and shown in Fig. 4.

$$
\begin{aligned}
S_1 &= [0,\alpha-\delta]\times[0,1], & T_1 &= S_1,\\
S_2 &= [\alpha-\delta,\alpha]\times[0,1], & T_2 &= [\alpha-\delta,\gamma]\times[0,1],\\
S_3 &= ([\alpha,\beta]\times[0,\gamma])\cup([\beta,\beta+\delta]\times[0,1]), & T_3 &= [\gamma,1]\times[0,\gamma],\\
S_4 &= [\beta+\delta,\beta+2\delta]\times[0,1], & T_4 &= ([\beta+2\delta,\gamma]\times[0,1])\cup([\gamma,1]\times[0,\gamma]),\\
S_5 &= [\beta+2\delta,\gamma-\delta]\times[0,1], & T_5 &= S_5,\\
S_6 &= [\gamma-\delta,\gamma]\times[0,1], & T_6 &= S_6,\\
S_7 &= [\gamma,1]\times[0,1] & T_7 &= [\gamma,1]\times[0,\gamma].
\end{aligned}
$$

The function g is a combination of 7 functions g_i, $1\le i\le 7$, each defined on S_i. Each function g_i is in turn a combination of some linear functions. For what we mean by linear functions on triangles and trapezoids, see the proof of Theorem 4.3. We describe each g_i as follows:

(1) Functions g_1 and g_5 are the identity function on S_1 and S_5, respectively.
(2) The function g_2 is linear on S_2 with the four corners mapped as follows: $g_2(\langle\alpha-\delta,0\rangle)=\langle\alpha-\delta,0\rangle$, $g_2(\langle\alpha,0\rangle)=\langle\gamma,0\rangle$, $g_2(\langle\alpha,1\rangle)=\langle\gamma,1\rangle$ and $g_2(\langle\alpha-\delta,1\rangle)=\langle\alpha-\delta,1\rangle$. That is, for any $\langle x,y\rangle$ in S_2, $g_2(\langle x,y\rangle)=\langle x',y\rangle$, where $\delta(\gamma-x')=(\gamma-\alpha+\beta)(a-x)$.
(3) The function g_3 is a combination of three linear functions. Let $u_1=\langle\alpha,\gamma\rangle$, $u_2=\langle\beta,\gamma\rangle$, $u_3=\langle\beta,1\rangle$, $u_4=\langle\beta+\delta,1\rangle$, $u_5=\langle\beta+\delta,0\rangle$, $u_6=\langle\alpha,0\rangle$, $v_1=\langle\gamma,\gamma\rangle$, $v_2=\langle 1-\delta',\gamma\rangle$, $v_3=\langle 1,\gamma\rangle$, $v_4=\langle 1,\gamma-\delta\rangle$, $v_5=\langle 1,0\rangle$ and $v_6=\langle\gamma,0\rangle$. The function g_3 maps each $u_i, 1\le i\le 6$, to v_i and is a linear function from trapezoid $S_{3,1}=\square u_1u_2u_5u_6$ to trapezoid $T_{3,1}=\square v_1v_2v_5v_6$, from triangle $S_{3,2}=\triangle u_2u_4u_5$ to triangle $T_{3,2}=\triangle v_2v_4v_5$, and from triangle $S_{3,3}=\triangle u_2u_3u_4$ to triangle $T_{3,3}=\triangle v_2v_3v_4$.



(a) Function g_3 (b) Function g_4

(c) Function g_6 (d) Function g_7

Fig. 5. Definitions of functions.

(4) The function g_4 is a combination of six linear functions. Let $u_1 = \langle \beta + \delta, 1\rangle$, $u_2 = \langle \beta + 9\delta/8, 1\rangle$, $u_3 = \langle \beta + 5\delta/4, 1\rangle$, $u_4 = \langle \beta + 11\delta/8, 1\rangle$, $u_5 = \langle \beta + 3\delta/2, 1\rangle$, $u_6 = \langle \beta + 2\delta, 1\rangle$, $u_7 = \langle \beta + 2\delta, 0\rangle$, $u_8 = \langle \beta + 3\delta/2, 0\rangle$, $u_9 = \langle \beta + \delta, 0\rangle$, $v_1 = \langle 1, \gamma - \delta\rangle$, $v_2 = \langle 1, \gamma\rangle$, $v_3 = \langle \gamma, \gamma\rangle$, $v_4 = \langle \gamma, 1\rangle$, $v_5 = \langle \gamma - \delta, 1\rangle$, $v_6 = u_6$, $v_7 = u_7$, $v_8 = \langle \gamma - \delta, 0\rangle$ and $v_9 = \langle 1, 0\rangle$. The function g_4 maps each u_i, $1 \le i \le 9$, to v_i and is a linear function from triangle $S_{4,1} = \triangle u_1u_8u_9$ to triangle $T_{4,1} = \triangle v_1v_8v_9$, from triangle $S_{4,2} = \triangle u_1u_2u_8$ to triangle $T_{4,2} = \triangle v_1v_2v_8$, from triangle $S_{4,3} = \triangle u_2u_3u_8$ to triangle $T_{4,3} = \triangle v_2v_3v_8$, from triangle $S_{4,4} = \triangle u_3u_4u_8$ to triangle $T_{4,4} = \triangle v_3v_4v_8$, from triangle $S_{4,5} = \triangle u_4u_5u_8$ to triangle $T_{4,5} = \triangle v_4v_5v_8$, and from rectangle $S_{4,6} = \square u_5u_6u_7u_8$ to rectangle $T_{4,6} = \square v_5v_6v_7v_8$.

(5) The function g_6 is a combination of three linear functions. Let $u_1 = \langle \gamma - \delta, 1\rangle$, $u_2 = \langle \gamma - \delta/2, 1\rangle$, $u_3 = \langle \gamma, 1\rangle$, $u_4 = \langle \gamma, 0\rangle$, $u_5 = \langle \gamma - \delta, 0\rangle$, $v_1 = u_1$, $v_2 = u_3$, $v_3 = \langle \gamma, \gamma\rangle$, $v_4 = u_4$ and $v_5 = u_5$. The function g_6 maps each u_i, $1 \le i \le 5$, to v_i and is a linear function from triangle $S_{6,1} = \triangle u_1u_2u_5$ to triangle $T_{6,1} = \triangle v_1v_2v_5$, from triangle $S_{6,2} = \triangle u_2u_3u_5$ to triangle $T_{6,2} = \triangle v_2v_3v_5$, and from triangle $S_{6,3} = \triangle u_3u_4u_5$ to triangle $T_{6,3} = \triangle v_3v_4v_5$.

(6) The function g_7 is a combination of two linear functions. Let $u_1 = \langle\gamma, 1\rangle$, $u_2 = \langle 1-\delta, 1\rangle$, $u_3 = \langle 1, 1\rangle$, $u_4 = \langle 1, 0\rangle$, $u_5 = \langle\gamma, 0\rangle$, $v_1 = \langle\gamma, \gamma\rangle$, $v_2 = \langle 1, \gamma\rangle$, $v_3 = u_4$, $v_4 = \langle 1-\delta, 0\rangle$ and $v_5 = u_5$. The function g_7 maps each u_i, $1 \le i \le 5$, to v_i and is a linear function from trapezoid $S_{7,1} = \Box u_1u_2u_4u_5$ to trapezoid $T_{7,1} = \Box v_1v_2v_4v_5$, and from triangle $S_{7,2} = \triangle u_2u_3u_4$ to triangle $T_{7,2} = \triangle v_2v_3v_4$.

The above definitions of functions g_3, g_4, g_6 and g_7 are shown in Fig. 5.

Similar to the function g constructed in the proof of Theorem 5.1, the function g is continuous and has the modulus of continuity $m(n) = n + 1 + \log(1/\delta')$. We observe that the regions T_1, T_2 and T_3, form a partition of the set T, that the regions T_5, T_6 and T_7 form a partition of the region T_4 and that T_4 is contained in $T_2 \cup T_3$. Since g is one-to-one on each region S_i, $1 \le i \le 7$, g is a three-to-one function on S. From the three-to-oneness of g and from the same argument for function g in Theorem 4.3, we can see that this g also has the polynomial inverse modulus of continuity $m(n)$. We leave it to the reader to verify it.

Now we are ready to define the function f. Let functions ψ, p and r, sequences a_n, b_n and γ_n, and sets W_n, V_n, and U_n be exactly the same as those defined in the proof of Theorem 4.3. Then define f_n in exactly the same way. That is, let g_n be the function g defined above with the parameters $\alpha_n = \psi(n) \cdot 2^{-p(n)}$, $\beta_n = (\psi(n)+1) \cdot 2^{-p(n)}$, $\delta_n = 2^{-(p(n)+2)}$ and $\delta'_n = 2^{-(p(n)+p(n+1)+3)}$; that is $g_n = g(\alpha_n, \beta_n, \delta_n, \delta'_n)$. Let $\gamma = 1 - (\beta_n - \alpha_n)$. We define the function f_n from U_n *onto* V_n by

$$f_n(\langle x, y\rangle) = \langle 1 - 2^{-r(n)}(1-u_n), 1 - 2^{-r(n)}(1-v_n)\rangle, \tag{5}$$

where

$$\langle u_n, v_n\rangle = g_n(\langle 2^{r(n)}(x-a_n), 2^{r(n)}(y-c_n)\rangle). \tag{6}$$

The function f is the combination of f_n on U_n, $n \ge 1$, plus $f(\langle x_0, 1\rangle) = \langle 1, 1\rangle$, where $x_0 = \lim a_n$. Note that $[0,1]^2 = \cup_{n=1}^{\infty} U_n \cup \{\langle x_0, 1\rangle\}$, and so f is defined on every point in $[0,1]^2$.

Similarly to the proof of Theorem 4.3, we can prove that f is well-defined on $[0,1]^2$, is polynomial-time computable and has a polynomial inverse modulus of continuity. We leave these proofs to the reader. We further observe that f maps $[0,1]^2$ *onto* $[0,1]^2$, because it maps each set U_n *onto* V_n and maps $\langle x_0, 1\rangle$, the only point in $[0,1]^2 - \cup_{n=1}^{\infty} U_n$, to $\langle 1, 1\rangle$, the only point in $[0,1]^2 - \cup_{n=1}^{\infty} V_n$. Finally, we observe that f is a three-to-one function, because each f_n is three-to-one and the range of f_n and the range of f_m, for $m \ne n$, intersect only on the boundary of V_n and V_m. The above completes the proof of the theorem. $\square$

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The author would like to thank Professor Osamu Watanabe for interesting discussions on one-way functions.

References

[1] E. Allender, O. Watanabe, Kolmogorov complexity and degrees of tally sets, in: Proc. Structure in Complexity Theory 3rd Annual Conf., 1988, pp. 102–111.
[2] J. Grollman, A. Selman, Complexity measures for public-key cryptosystems, in: Proc. 25th IEEE Symp. on Foundations of Computer Science, 1984, pp. 495–503.
[3] K. Ko, On some natural complete operators, Theor. Comput. Sci. 37 (1985) 1–30.
[4] K. Ko, Applying techniques of discrete complexity theory to numerical computation, in: R. Book (Ed.), Studies in Complexity Theory, Pitman, London, 1986.
[5] K. Ko, I.I. Friedman, Computational complexity of real functions, Theor. Comput. Sci. 20 (1982) 323–352.
[6] L. Valiant, Relative complexity of checking and evaluating, Inf. Process. Lett. 5 (1976) 20–23.
[7] O. Watanabe, On hardness of one-way functions, Inf. Process. Lett. 27 (1988) 151–157.