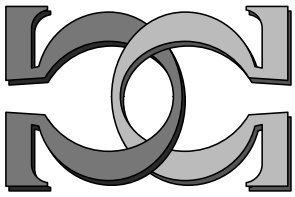
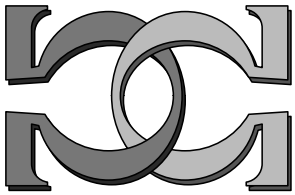
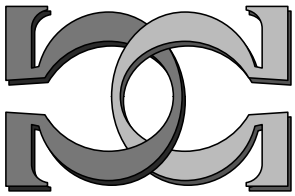
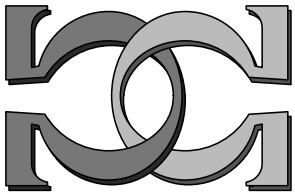


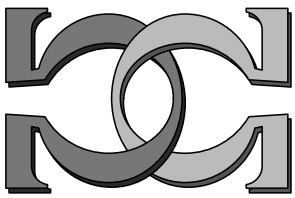




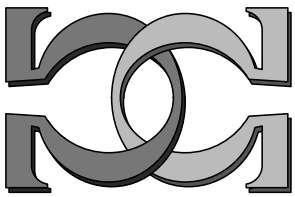
CDMTCS Research Report Series



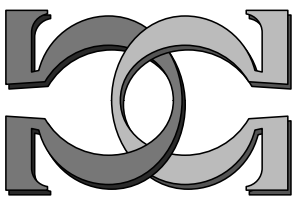
Chaitin Ω numbers and halting problems



Kohtaro Tadaki
Chuo University, Japan



CDMTCS-359
April 2009



Centre for Discrete Mathematics and
Theoretical Computer Science

Chaitin Ω numbers and halting problems

Kohtaro Tadaki

Research and Development Initiative, Chuo University
CREST, JST
1–13–27 Kasuga, Bunkyo-ku, Tokyo 112-8551, Japan
E-mail: tadaki@kc.chuo-u.ac.jp
http://www2.odn.ne.jp/tadaki/

Abstract. Chaitin [G. J. Chaitin, *J. Assoc. Comput. Mach.*, vol. 22, pp. 329–340, 1975] introduced Ω number as a concrete example of random real. The real Ω is defined as the probability that an optimal computer halts, where the optimal computer is a universal decoding algorithm used to define the notion of program-size complexity. Chaitin showed Ω to be random by discovering the property that the first n bits of the base-two expansion of Ω solve the halting problem of the optimal computer for all binary inputs of length at most n. In the present paper we investigate this property from various aspects. We consider the relative computational power between the base-two expansion of Ω and the halting problem by imposing the restriction to finite size on both the problems. It is known that the base-two expansion of Ω and the halting problem are Turing equivalent. We thus consider an elaboration of the Turing equivalence in a certain manner.

Key words: algorithmic information theory, Chaitin Ω number, halting problem, Turing equivalence, algorithmic randomness, program-size complexity

1 Introduction

Algorithmic information theory (AIT, for short) is a framework for applying information-theoretic and probabilistic ideas to recursive function theory. One of the primary concepts of AIT is the *program-size complexity* (or *Kolmogorov complexity*) $H(s)$ of a finite binary string s, which is defined as the length of the shortest binary input for a universal decoding algorithm U, called an *optimal computer*, to output s. By the definition, $H(s)$ can be thought of as the information content of the individual finite binary string s. In fact, AIT has precisely the formal properties of classical information theory (see Chaitin [2]). In particular, the notion of program-size complexity plays a crucial role in characterizing the *randomness* of an infinite binary string, or equivalently, a real. In [2] Chaitin introduced the halting probability Ω_U as an example of random real. His Ω_U is defined as the probability that the optimal computer U halts, and plays a central role in the metamathematical development of AIT. The real Ω_U is shown to be random, based on the following fact:[1]

Fact 1 (Chaitin [2])**.** *The first n bits of the base-two expansion of Ω_U solve the halting problem of U for inputs of length at most n.* □

[1] A rigorous form of Fact 1 is seen in Theorem 5.3 below in a more general form.

In this paper, we first consider the following converse problem:

Problem 1. *For every positive integer n, if n and the list of all halting inputs for U of length at most n are given, can the first n bits of the base-two expansion of Ω_U be calculated ?* □

As a result of this paper, we can answer this problem negatively. In this paper, however, we consider more general problems in the following forms. Let V and W be optimal computers.

Problem 2. *Find a succinct equivalent characterization of a total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$ which satisfies the condition: For all $n \in \mathbb{N}^+$, if n and the list of all halting inputs for V of length at most n are given, then the first $n - f(n) - O(1)$ bits of the base-two expansion of Ω_W can be calculated.* □

Problem 3. *Find a succinct equivalent characterization of a total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$ which satisfies the condition: For infinitely many $n \in \mathbb{N}^+$, if n and the list of all halting inputs for V of length at most n are given, then the first $n - f(n) - O(1)$ bits of the base-two expansion of Ω_W can be calculated.* □

Here $\mathbb{N}^+$ denotes the set of positive integers and $\mathbb{N} = \{0\} \cup \mathbb{N}^+$. Theorem 3.1 and Theorem 4.1 below are two of the main results of this paper. On the one hand, Theorem 3.1 gives to Problem 2 a solution that the total recursive function f must satisfy $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$, which is the Kraft inequality in essence. Note that the condition $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$ holds for $f(n) = \lfloor (1+\varepsilon)\log_2 n \rfloor$ with an arbitrary computable real $\varepsilon > 0$, while this condition does not hold for $f(n) = \lfloor \log_2 n \rfloor$. On the other hand, Theorem 4.1 gives to Problem 3 a solution that the total recursive function f must not be bounded to the above. Theorem 4.1 also results in Corollary 4.2 below, which refutes Problem 1 completely.

It is also important to consider whether the bound n on the length of halting inputs given in Fact 1 is tight or not. We consider this problem in the following form:

Problem 4. *Find a succinct equivalent characterization of a total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$ which satisfies the condition: For all $n \in \mathbb{N}^+$, if n and the first n bits of the base-two expansion of Ω_V are given, then the list of all halting inputs for W of length at most $n + f(n) - O(1)$ can be calculated.* □

Theorem 5.1, which is one of the main results of this paper, gives to Problem 4 a solution that the total recursive function f must be bounded to the above. Thus, we see that the bound n on the length of halting inputs given in Fact 1 is tight up to an additive constant.

It is well known that the base-two expansion of Ω_U and the halting problem of U are Turing equivalent, i.e., $\Omega_U \equiv_T \mathrm{dom}\, U$ holds, where $\mathrm{dom}\, U$ denotes the domain of definition of U. This paper investigates an elaboration of the Turing equivalence. For example, consider the Turing reduction $\Omega_U \leq_T \mathrm{dom}\, U$, which partly constitutes the Turing equivalence $\Omega_U \equiv_T \mathrm{dom}\, U$. The Turing reduction can be equivalent to the condition that there exists an oracle deterministic Turing machine M such that, for all $n \in \mathbb{N}^+$,

$$M^{\mathrm{dom}\, U}(n) = \Omega_U{\restriction}_n, \tag{1}$$

where $\Omega_U{\restriction}_n$ denotes the first n bits of the base-two expansion of Ω_U. Let $g\colon \mathbb{N}^+ \to \mathbb{N}$ and $h\colon \mathbb{N}^+ \to \mathbb{N}$ be total recursive functions. Then the condition (1) can be elaborated to the condition that there exists an oracle deterministic Turing machine M such that, for all $n \in \mathbb{N}^+$,

$$M^{\mathrm{dom}\, U{\restriction}_{g(n)}}(n) = \Omega_U{\restriction}_{h(n)}, \tag{2}$$

where $\mathrm{dom}\, U\restriction_{g(n)}$ denotes the set of all strings in $\mathrm{dom}\, U$ of length at most $g(n)$. This elaboration allows us to consider the asymptotic behavior of h which satisfies the condition (2), for a given g. We might regard g as the degree of the relaxation of the restrictions on the computational resource (i.e., on the oracle $\mathrm{dom}\, U$) and h as the difficulty of the problem to solve. Thus, even in the context of computability theory, we can deal with the notion of asymptotic behavior in a manner like in computational complexity theory in some sense. Theorem 3.1, a solution to Problem 2, is obtained as a result of the investigation in this line, and gives the upper bound of the function h in the case of $g(n) = n$.

The other Turing reduction $\mathrm{dom}\, U \leq_T \Omega_U$, which constitutes $\Omega_U \equiv_T \mathrm{dom}\, U$, is also elaborated in the same manner as above to lead to Theorem 5.1, a solution to Problem 4.

Thus, in this paper, we study the relationship between the base-two expansion of Ω and the halting problem of an optimal computer using a more rigorous and insightful notion than the notion of Turing equivalence. The paper is organized as follows. We begin in Section 2 with some preliminaries to AIT. We then prove Theorems 3.1, 4.1, and 5.1 in Sections 3, 4, and 5, respectively.

2 Preliminaries

2.1 Basic notation

We start with some notation about numbers and strings which will be used in this paper. $\#S$ is the cardinality of S for any set S. $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ is the set of natural numbers, and $\mathbb{N}^+$ is the set of positive integers. $\mathbb{Z}$ is the set of integers, and $\mathbb{Q}$ is the set of rational numbers. $\mathbb{R}$ is the set of real numbers. Let $f\colon S \to \mathbb{R}$ with $S \subset \mathbb{R}$. We say that f is *increasing* (resp., *non-decreasing*) if $f(x) < f(y)$ (resp., $f(x) \leq f(y)$) for all $x, y \in S$ with $x < y$.

Normally, $O(1)$ denotes any function $f\colon \mathbb{N}^+ \to \mathbb{R}$ such that there is $C \in \mathbb{R}$ with the property that $|f(n)| \leq C$ for all $n \in \mathbb{N}^+$.

$\{0,1\}^* = \{\lambda, 0, 1, 00, 01, 10, 11, 000, \dots\}$ is the set of finite binary strings where λ denotes the *empty string*, and $\{0,1\}^*$ is ordered as indicated. We identify any string in $\{0,1\}^*$ with a natural number in this order, i.e., we consider $\varphi\colon \{0,1\}^* \to \mathbb{N}$ such that $\varphi(s) = 1s - 1$ where the concatenation $1s$ of strings 1 and s is regarded as a dyadic integer, and then we identify s with $\varphi(s)$. For any $s \in \{0,1\}^*$, $|s|$ is the *length* of s. A subset S of $\{0,1\}^*$ is called *prefix-free* if no string in S is a prefix of another string in S. For any subset S of $\{0,1\}^*$ and any $n \in \mathbb{Z}$, we denote by $S\restriction_n$ the set $\{s \in S \mid |s| \leq n\}$. Note that $S\restriction_n = \emptyset$ for every subset S of $\{0,1\}^*$ and every negative integer $n \in \mathbb{Z}$. $\{0,1\}^\infty$ is the set of infinite binary strings, where an infinite binary string is infinite to the right but finite to the left. For any partial function f, the domain of definition of f is denoted by $\mathrm{dom}\, f$. We write "r.e." instead of "recursively enumerable."

Let α be an arbitrary real number. $\lfloor \alpha \rfloor$ is the greatest integer less than or equal to α, and $\lceil \alpha \rceil$ is the smallest integer greater than or equal to α. For any $n \in \mathbb{N}^+$, we denote by $\alpha\restriction_n \in \{0,1\}^*$ the first n bits of the base-two expansion of $\alpha - \lfloor \alpha \rfloor$ with infinitely many zeros. For example, in the case of $\alpha = 5/8$, $\alpha\restriction_6 = 101000$. On the other hand, for any non-positive integer $n \in \mathbb{Z}$, we set $\alpha\restriction_n = \lambda$.

A real number α is called *r.e.* if there exists a total recursive function $f\colon \mathbb{N}^+ \to \mathbb{Q}$ such that $f(n) \leq \alpha$ for all $n \in \mathbb{N}^+$ and $\lim_{n\to\infty} f(n) = \alpha$. An r.e. real number is also called a *left-computable* real number.

2.2 Algorithmic information theory

In the following we concisely review some definitions and results of algorithmic information theory [2, 4]. A *computer* is a partial recursive function $C\colon \{0,1\}^* \to \{0,1\}^*$ such that $\operatorname{dom} C$ is a prefix-free set. For each computer C and each $s \in \{0,1\}^*$, $H_C(s)$ is defined by $H_C(s) = \min\{\, |p| \mid p \in \{0,1\}^* \,\&\, C(p) = s \,\}$ (may be ∞). A computer U is said to be *optimal* if for each computer C there exists $d \in \mathbb{N}$ with the following property; if $p \in \operatorname{dom} C$, then there is q for which $U(q) = C(p)$ and $|q| \leq |p| + d$. It is easy to see that there exists an optimal computer. We choose a particular optimal computer U as the standard one for use, and define $H(s)$ as $H_U(s)$, which is referred to as the *program-size complexity* of s, the *information content* of s, or the *Kolmogorov complexity* of s [7, 9, 2]. It follows that for every computer C there exists $d \in \mathbb{N}$ such that, for every $s \in \{0,1\}^*$,

$$H(s) \leq H_C(s) + d. \tag{3}$$

Based on this we can show that there exists $c \in \mathbb{N}$ such that, for every $s \in \{0,1\}^*$,

$$H(s) \leq 2\,|s| + c. \tag{4}$$

Using (3) we can also show that, for every partial recursive function $\Psi\colon \{0,1\}^* \to \{0,1\}^*$, there exists $c \in \mathbb{N}$ such that, for every $s \in \operatorname{dom}\Psi$,

$$H(\Psi(s)) \leq H(s) + c. \tag{5}$$

For any $s \in \{0,1\}^*$, we define s^* as $\min\{\, p \in \{0,1\}^* \mid U(p) = s\}$, i.e., the first element in the ordered set $\{0,1\}^*$ of all strings p such that $U(p) = s$. Then, $|s^*| = H(s)$ for every $s \in \{0,1\}^*$. For any $s, t \in \{0,1\}^*$, we define $H(s,t)$ as $H(b(s,t))$, where $b\colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ is a particular bijective total recursive function. Note also that, for every $n \in \mathbb{N}$, $H(n)$ is $H(\text{the } n\text{th element of } \{0,1\}^*)$.

Definition 2.1 (Chaitin Ω number, Chaitin [2])**.** *For any optimal computer V, the halting probability Ω_V of V is defined by*

$$\Omega_V = \sum_{p \in \operatorname{dom} V} 2^{-|p|}.$$

□

For every optimal computer V, since $\operatorname{dom} V$ is prefix-free, Ω_V converges and $0 < \Omega_V \leq 1$. For any $\alpha \in \mathbb{R}$, we say that α is *weakly Chaitin random* if there exists $c \in \mathbb{N}$ such that $n - c \leq H(\alpha\restriction_n)$ for all $n \in \mathbb{N}^+$ [2, 4].

Theorem 2.2 (Chaitin [2])**.** *For every optimal computer V, Ω_V is weakly Chaitin random.* □

Therefore $0 < \Omega_V < 1$ for every optimal computer V. For any $\alpha \in \mathbb{R}$, we say that α is *Chaitin random* if $\lim_{n\to\infty} H(\alpha\restriction_n) - n = \infty$ [2, 4]. We can then show the following theorem (see Chaitin [4] for the proof and historical detail).

Theorem 2.3. *For every $\alpha \in \mathbb{R}$, α is weakly Chaitin random if and only if α is Chaitin random.* □

The following is an important result on random r.e. reals.

Theorem 2.4 (Calude, et al. [1], Kučera and Slaman [8])**.** *For every $\alpha \in (0,1)$, α is r.e. and weakly Chaitin random if and only if there exists an optimal computer V such that $\alpha = \Omega_V$.* □

3 Elaboration I of the Turing reduction $\Omega_U \leq_T \operatorname{dom} U$

Theorem 3.1 (main result I)**.** *Let V and W be optimal computers, and let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. Then the following two conditions are equivalent:*

(i) There exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_n}(n) = \Omega_W\restriction_{n-f(n)-c}$.

(ii) $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$. □

Theorem 3.1 follows from Theorem 3.2 and Theorem 3.3 below, and Theorem 2.4.

Theorem 3.2. *Let α be an r.e. real, and let V be an optimal computer. For every total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, if $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$, then there exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_n}(n) = \alpha\restriction_{n-f(n)-c}$.* □

Theorem 3.3. *Let α be a real which is weakly Chaitin random, and let V be an optimal computer. For every total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, if there exists an oracle deterministic Turing machine M such that, for all $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_n}(n) = \alpha\restriction_{n-f(n)}$, then $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$.* □

The proofs of Theorem 3.2 and Theorem 3.3 are given in the next two subsections, respectively.

Note that, as a variant of Theorem 3.1, we can prove the following theorem as well, in a similar manner to the proof of Theorem 3.1.

Theorem 3.4 (variant of the main result I)**.** *Let V and W be optimal computers, and let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. Then the following two conditions are equivalent:*

(i) There exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_{n+f(n)+c}}(n) = \Omega_W\restriction_n$.

(ii) $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$. □

3.1 The proof of Theorem 3.2

In order to prove Theorem 3.2, we need Theorem 3.5 and Corollary 3.8 below.

Theorem 3.5 (Kraft-Chaitin Theorem, Chaitin [2])**.** *Let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function such that $\sum_{n=1}^{\infty} 2^{-f(n)} \leq 1$. Then there exists a total recursive function $g\colon \mathbb{N}^+ \to \{0,1\}^*$ such that (i) the function g is an injection, (ii) the set $\{\, g(n) \mid n \in \mathbb{N}^+ \,\}$ is prefix-free, and (iii) $|g(n)| = f(n)$ for all $n \in \mathbb{N}^+$.* □

Let M be a deterministic Turing machine with the input and output alphabet $\{0,1\}$, and let C be a computer. We say that M *computes* C if the following holds: for every $p \in \{0,1\}^*$, when M starts with the input p, (i) M halts and outputs $C(p)$ if $p \in \operatorname{dom} C$; (ii) M does not halt forever otherwise. We use this convention on the computation of a computer by a deterministic Turing machine throughout the rest of this paper. Thus, we exclude the possibility that there is $p \in \{0,1\}^*$ such that, when M starts with the input p, M halts but $p \notin \operatorname{dom} C$.

Theorem 3.6. *Let V be an optimal computer. Then, for every computer C there exists $d \in \mathbb{N}$ such that, for every $p \in \{0,1\}^*$, if p and the list of all halting inputs for V of length at most $|p| + d$ are given, then the halting problem of the input p for C can be solved.*

Proof. Let M be a deterministic Turing machine which computes a computer C. For each $p \in \{0,1\}^*$, let $h_M(p)$ be the computation history of M from the initial configuration with input p, and let $\mathrm{bin}_M(p) \in \{0,1\}^* \cup \{0,1\}^\infty$ be the binary representation of $h_M(p)$ in a certain format. Note that $\mathrm{bin}_M(p) \in \{0,1\}^*$ if and only if $p \in \mathrm{dom}\, C$ for every $p \in \{0,1\}^*$, by our convention on the computation of a computer by a deterministic Turing machine. We consider the computer D such that (i) $\mathrm{dom}\, D = \mathrm{dom}\, C$ and (ii) $D(p) = \mathrm{bin}_M(p)$ for every $p \in \mathrm{dom}\, C$. It is easy to see that such a computer D exists. Then, since V is an optimal computer, from the definition of optimality there exists $d \in \mathbb{N}$ with the following property; if $p \in \mathrm{dom}\, D$, then there is q for which $V(q) = D(p)$ and $|q| \leq |p| + d$.

Given $p \in \{0,1\}^*$ and the list $\{q_1, \ldots, q_L\}$ of all halting inputs for V of length at most $|p|+d$, one first calculates the finite set $S_p = \{\, V(q_i) \mid i = 1, \ldots, L \,\}$. One then checks whether $\mathrm{bin}_M(p) \in S_p$ or not. This can be possible since S_p is a finite subset of $\{0,1\}^*$. In the case of $\mathrm{bin}_M(p) \in S_p$, $\mathrm{bin}_M(p) \in \{0,1\}^*$ and therefore $p \in \mathrm{dom}\, C$. On the other hand, if $p \in \mathrm{dom}\, C$, then there is q such that $V(q) = \mathrm{bin}_M(p)$ and $|q| \leq |p| + d$, and therefore $q \in \{q_1, \ldots, q_L\}$ and $\mathrm{bin}_M(p) \in S_p$. Thus, $p \notin \mathrm{dom}\, C$ in the case of $\mathrm{bin}_M(p) \notin S_p$. □

Remark 3.7. *A partial recursive function $u\colon \{0,1\}^* \to \{0,1\}^*$ is called optimal if for every partial recursive function $f\colon \{0,1\}^* \to \{0,1\}^*$ there exists $d \in \mathbb{N}$ such that, for every $p \in \mathrm{dom}\, f$, there is $q \in \mathrm{dom}\, u$ for which $u(q) = f(p)$ and $|q| \leq |p| + d$. An optimal partial recursive function $u\colon \{0,1\}^* \to \{0,1\}^*$ is used to define the notion of plain program-size complexity. Obviously, we can show that the same theorem as Theorem 3.6 holds between the halting problem of any optimal partial recursive function $u\colon \{0,1\}^* \to \{0,1\}^*$ and one of any partial recursive function $f\colon \{0,1\}^* \to \{0,1\}^*$.* □

As a corollary of Theorem 3.6 above we obtain the following.

Corollary 3.8. *Let V be an optimal computer. Then, for every computer C there exist an oracle deterministic Turing machine M and $d \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\mathrm{dom}\, V\restriction_{n+d}}(n) = \mathrm{dom}\, C\restriction_n$, where the finite subset $\mathrm{dom}\, C\restriction_n$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format.* □

Based on Theorem 3.5 and Corollary 3.8, Theorem 3.2 is proved as follows.

Proof of Theorem 3.2. Let α be an r.e. real, and let V be an optimal computer. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$. In the case of $\alpha \in \mathbb{Q}$, the result is obvious. Thus, in what follows, we assume that $\alpha \notin \mathbb{Q}$ and therefore the base-two expansion of $\alpha - \lfloor \alpha \rfloor$ is unique and contains infinitely many ones.

Since $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$, there exists $d_0 \in \mathbb{N}$ such that $\sum_{n=1}^{\infty} 2^{-f(n)-d_0} \leq 1$. Hence, by the Kraft-Chaitin Theorem, i.e., Theorem 3.5, there exists a total recursive function $g\colon \mathbb{N}^+ \to \{0,1\}^*$ such that (i) the function g is an injection, (ii) the set $\{\, g(n) \mid n \in \mathbb{N}^+ \}$ is prefix-free, and (iii) $|g(n)| = f(n) + d_0$ for all $n \in \mathbb{N}^+$. On the other hand, since α is r.e., there exists a total recursive function $h\colon \mathbb{N}^+ \to \mathbb{Q}$ such that $h(k) \leq \alpha$ for all $k \in \mathbb{N}^+$ and $\lim_{k\to\infty} h(k) = \alpha$.

Now, let us consider the following computer C. For each $n \in \mathbb{N}^+$, $p, s \in \{0,1\}^*$ and $l \in \mathbb{N}$ such that $U(p) = l$, $g(n)ps \in \mathrm{dom}\, C$ if and only if (i) $|g(n)ps| = n - l$ and (ii) $0.s < h(k) - \lfloor \alpha \rfloor$ for some $k \in \mathbb{N}^+$. It is easy to see that such a computer C exists. Then, by Corollary 3.8, there exist an oracle deterministic Turing machine M and $d \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\mathrm{dom}\, V\restriction_{n+d}}(n) = \mathrm{dom}\, C\restriction_n$,

where the finite subset $\mathrm{dom}\, C\restriction_n$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format. We then see that, for every $n \in \mathbb{N}^+$ and $s \in \{0,1\}^*$ such that $|s| = n - |g(n)| - d - |d^*|$,

$$g(n)d^*s \in \mathrm{dom}\, C \text{ if and only if } s \leq \alpha\restriction_{n-|g(n)|-d-|d^*|}, \tag{6}$$

where s and $\alpha\restriction_{n-|g(n)|-d-|d^*|}$ are regarded as a dyadic integer. Then, by the following procedure, we see that there exist an oracle deterministic Turing machine M_1 and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M_1^{\mathrm{dom}\, V\restriction_n}(n) = \alpha\restriction_{n-f(n)-c}$. Note here that $|g(n)| = f(n) + d_0$ for all $n \in \mathbb{N}^+$ and also $H(d) = |d^*|$.

Given n and $\mathrm{dom}\, V\restriction_n$ with $n > d$, one first checks whether $n - |g(n)| - d - H(d) \leq 0$ holds. If this holds then one outputs λ. If this does not hold, one then calculates the finite set $\mathrm{dom}\, C\restriction_{n-d}$ by simulating the computation of M with the input $n-d$ and the oracle $\mathrm{dom}\, V\restriction_n$. Then, based on (6), one determines $\alpha\restriction_{n-|g(n)|-d-H(d)}$ by checking whether $g(n)d^*s \in \mathrm{dom}\, C$ holds or not for each $s \in \{0,1\}^*$ with $|s| = n - |g(n)| - d - H(d)$. This is possible since $|g(n)d^*s| = n - d$ for every $s \in \{0,1\}^*$ with $|s| = n - |g(n)| - d - H(d)$. Finally, one outputs $\alpha\restriction_{n-|g(n)|-d-H(d)}$. □

3.2 The proof of Theorem 3.3

In order to prove Theorem 3.3, we need Theorem 3.9 and the Ample Excess Lemma (i.e., Theorem 3.10) below.

Let M be an arbitrary deterministic Turing machine with the input alphabet $\{0,1\}$. We define $L_M = \min\{\, |p| \mid p \in \{0,1\}^* \;\&\; M \text{ halts on input } p \,\}$ (may be ∞). For any $n \geq L_M$, we define T_n^M as the maximum running time of M on all halting inputs of length at most n.

Theorem 3.9. *Let V be an optimal computer, and let M be a deterministic Turing machine which computes V. Then $n = H(T_n^M, n) + O(1) = H(T_n^M) + O(1)$ for all $n \geq L_M$.* □

Note that Solovay [14] showed a similar result to Theorem 3.9 for $h_n = \#\{p \in \mathrm{dom}\, V \mid |p| \leq n\}$ in place of T_n^M. On the other hand, Chaitin showed a similar result to Theorem 3.9 for $p \in \mathrm{dom}\, V$ such that $|p| \leq n$ and the running time of M on the input p equals to T_n^M, in place of T_n^M (see Note in Section 8.1 of Chaitin [4]). We include the proof of Theorem 3.9 in Appendix A for completeness.

Miller and Yu [11] recently strengthened Theorem 2.3 to the following form.

Theorem 3.10 (Ample Excess Lemma, Miller and Yu [11])**.** *For every $\alpha \in \mathbb{R}$, α is weakly Chaitin random if and only if $\sum_{n=1}^{\infty} 2^{n-H(\alpha\restriction_n)} < \infty$.* □

Then the proof of Theorem 3.3 is as follows.

Proof of Theorem 3.3. Let α be a real which is weakly Chaitin random. Let V be an optimal computer, and let M be a deterministic Turing machine which computes V. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that there exists an oracle deterministic Turing machine M_0 such that, for all $n \in \mathbb{N}^+$, $M_0^{\mathrm{dom}\, V\restriction_n}(n) = \alpha\restriction_{n-f(n)}$. Note that, given (T_n^M, n) with $n \geq L_M$, one can calculate the finite set $\mathrm{dom}\, V\restriction_n$ by simulating the computation of M with the input p until at most T_n^M steps, for each $p \in \{0,1\}^*$ with $|p| \leq n$. Thus, we see that there exists a partial recursive function $\Psi\colon \mathbb{N} \times \mathbb{N}^+ \to \{0,1\}^*$ such that, for all $n \geq L_M$, $\Psi(T_n^M, n) = \alpha\restriction_{n-f(n)}$. It follows from (5) that $H(\alpha\restriction_{n-f(n)}) \leq H(T_n^M, n) + O(1)$ for all $n \geq L_M$. Thus, by Theorem 3.9 we have

$$H(\alpha\restriction_{n-f(n)}) \leq n + O(1) \tag{7}$$

for all $n \in \mathbb{N}^+$.

In the case where the function $n - f(n)$ of n is bounded to the above, there exists $c \in \mathbb{N}$ such that, for every $n \in \mathbb{N}^+$, $-f(n) \leq c - n$, and therefore $\sum_{n=1}^{\infty} 2^{-f(n)} \leq 2^c$. Thus, in what follows, we assume that the function $n - f(n)$ of n is not bounded to the above.

We define a function $g\colon \mathbb{N}^+ \to \mathbb{Z}$ by $g(n) = \max\{k - f(k) \mid 1 \leq k \leq n\}$. It follows that the function g is non-decreasing and $\lim_{n\to\infty} g(n) = \infty$. Thus we can choose an enumeration $n_1, n_2, n_3, \ldots$ of the countably infinite set $\{n \in \mathbb{N}^+ \mid n \geq 2 \ \&\ 0 \leq g(n-1) < g(n)\}$ with $n_j < n_{j+1}$. It is then easy to see that $g(n_j) = n_j - f(n_j)$ and $1 \leq n_j - f(n_j) < n_{j+1} - f(n_{j+1})$ hold for all j. On the other hand, since α is weakly Chaitin random, using the Ample Excess Lemma, i.e., Theorem 3.10, we have $\sum_{n=1}^{\infty} 2^{n-H(\alpha\restriction_n)} < \infty$. Thus, using (7) we see that

$$\sum_{j=1}^{\infty} 2^{-f(n_j)} \leq \sum_{j=1}^{\infty} 2^{n_j - f(n_j) - H(\alpha\restriction_{n_j - f(n_j)}) + O(1)} \leq \sum_{n=1}^{\infty} 2^{n - H(\alpha\restriction_n) + O(1)} < \infty. \tag{8}$$

On the other hand, it is easy to see that (i) $g(n) \geq n - f(n)$ for every $n \in \mathbb{N}^+$, and (ii) $g(n) = g(n_j)$ for every j and n with $n_j \leq n < n_{j+1}$. Thus, for each $k \geq 2$, it is shown that

$$\begin{aligned}\sum_{n=n_1}^{n_k-1} 2^{-f(n)} &\leq \sum_{n=n_1}^{n_k-1} 2^{g(n)-n} = \sum_{j=1}^{k-1} \sum_{n=n_j}^{n_{j+1}-1} 2^{g(n)-n} = \sum_{j=1}^{k-1} 2^{g(n_j)} \sum_{n=n_j}^{n_{j+1}-1} 2^{-n} \\ &= \sum_{j=1}^{k-1} 2^{n_j - f(n_j)} 2^{-n_j+1} \left(1 - 2^{-n_{j+1}+n_j}\right) < 2 \sum_{j=1}^{k-1} 2^{-f(n_j)}.\end{aligned}$$

Thus, using (8) we see that $\lim_{k\to\infty} \sum_{n=n_1}^{n_k-1} 2^{-f(n)} < \infty$. Since $2^{-f(n)} > 0$ for all $n \in \mathbb{N}^+$ and $\lim_{j\to\infty} n_j = \infty$, we have $\sum_{n=1}^{\infty} 2^{-f(n)} < \infty$. □

4 Elaboration II of the Turing reduction $\Omega_U \leq_T \operatorname{dom} U$

Theorem 4.1 (main result II)**.** *Let V and W be optimal computers, and let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. Then the following two conditions are equivalent:*

(i) There exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for infinitely many $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_n}(n) = \Omega_W\restriction_{n-f(n)-c}$.

(ii) The function f is not bounded to the above. □

The proof of Theorem 4.1 is given in Subsection 4.1 below. By setting $f(n) = 0$ and $W = V$ in Theorem 4.1, we obtain the following.

Corollary 4.2. *Let V be an optimal computer. Then, for every $c \in \mathbb{N}$, there does not exist an oracle deterministic Turing machine M such that, for infinitely many $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_{n+c}}(n) = \Omega_V\restriction_n$.* □

Note that, as a variant of Theorem 4.1, we can prove the following theorem as well, in a similar manner to the proof of Theorem 4.1.

Theorem 4.3 (variant of the main result II)**.** *Let V and W be optimal computers, and let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. Then the following two conditions are equivalent:*

(i) There exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for infinitely many $n \in \mathbb{N}^+$, $M^{\mathrm{dom}\, V\restriction_{n+f(n)+c}}(n) = \Omega_W\restriction_n$.

(ii) The function f is not bounded to the above. □

4.1 The proof of Theorem 4.1

Theorem 4.1 follows from Theorem 4.4 and Theorem 4.6 below, and Theorem 2.4.

Theorem 4.4. *Let α be an r.e. real, and let V be an optimal computer. For every total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, if the function f is not bounded to the above, then there exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for infinitely many $n \in \mathbb{N}^+$, $M^{\mathrm{dom}\, V\restriction_n}(n) = \alpha\restriction_{n-f(n)-c}$.* □

In order to prove Theorem 4.4, we need Lemma 4.5 below. It is easy to show Lemma 4.5. For completeness, however, we include the proof of Lemma 4.5 in Appendix B.

Lemma 4.5. *Let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. If the function f is not bounded to the above, then $H(n) \leq f(n)$ for infinitely many $n \in \mathbb{N}^+$.* □

Based on Lemma 4.5 and Corollary 3.8, Theorem 4.4 is proved as follows.

Proof of Theorem 4.4. Let α be an r.e. real, and let V be an optimal computer. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that the function f is not bounded to the above. In the case of $\alpha \in \mathbb{Q}$, the result is obvious. Thus, in what follows, we assume that $\alpha \notin \mathbb{Q}$ and therefore the base-two expansion of $\alpha - \lfloor \alpha \rfloor$ is unique and contains infinitely many ones.

Since the total recursive function f is not bounded to the above, by Lemma 4.5 we see that $H(n) \leq f(n)$ for infinitely many $n \in \mathbb{N}^+$. Note also that $\lim_{n\to\infty} n - H(n) = \infty$. This is because $H(n) \leq 2\log_2 n + O(1)$ holds for all $n \in \mathbb{N}^+$ by (4). On the other hand, since α is r.e., there exists a total recursive function $g\colon \mathbb{N}^+ \to \mathbb{Q}$ such that $g(k) \leq \alpha$ for all $k \in \mathbb{N}^+$ and $\lim_{k\to\infty} g(k) = \alpha$.

Let us consider the following computer C. For each $p, q, s \in \{0,1\}^*$ and $n, l \in \mathbb{N}$ such that $U(p) = n$ and $U(q) = l$, $pqs \in \mathrm{dom}\, C$ if and only if (i) $|pqs| = n - l$ and (ii) $0.s < g(k) - \lfloor \alpha \rfloor$ for some $k \in \mathbb{N}^+$. It is easy to see that such a computer C exists. Then, by Corollary 3.8, there exist an oracle deterministic Turing machine M and $d \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\mathrm{dom}\, V\restriction_{n+d}}(n) = \mathrm{dom}\, C\restriction_n$, where the finite subset $\mathrm{dom}\, C\restriction_n$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format. We then see that, for every $n \in \mathbb{N}^+$ and $p, s \in \{0,1\}^*$ such that $U(p) = n$ and $|s| = n - |p| - d - |d^*|$,

$$pd^*s \in \mathrm{dom}\, C \text{ if and only if } s \leq \alpha\restriction_{n-|p|-d-|d^*|}, \tag{9}$$

where s and $\alpha\restriction_{n-|p|-d-|d^*|}$ are regarded as a dyadic integer. Then, by the following procedure, we see that there exist an oracle deterministic Turing machine M_1 and $c \in \mathbb{N}$ such that, for infinitely many $n \in \mathbb{N}^+$, $M_1^{\mathrm{dom}\, V\restriction_n}(n) = \alpha\restriction_{n-f(n)-c}$. Note here that $H(d) = |d^*|$.

Given n and $\mathrm{dom}\, V\restriction_n$ with $n > d$, one first tries to find $p \in \{0,1\}^*$ which satisfies that (i) $U(p) = n$, (ii) $|p| \leq f(n)$, and (iii) $n - |p| - d - H(d) \geq 1$. One can find such a string p for the cases of infinitely many $n \in \mathbb{N}^+$. This is because $H(k) \leq f(k)$ holds for infinitely many $k \in \mathbb{N}^+$ and $\lim_{k\to\infty} k - H(k) = \infty$. If such a string p is found, one then calculates the finite set $\mathrm{dom}\, C\restriction_{n-d}$ by simulating the computation of M with the input $n - d$ and the oracle $\mathrm{dom}\, V\restriction_n$. Then, based

on (9), one determines $\alpha\restriction_{n-|p|-d-H(d)}$ by checking whether $pd^*s \in \operatorname{dom} C$ holds or not for each $s \in \{0,1\}^*$ with $|s| = n-|p|-d-H(d)$. This is possible since $|pd^*s| = n-d$ for every $s \in \{0,1\}^*$ with $|s| = n-|p|-d-H(d)$. Finally, one calculates and outputs $\alpha\restriction_{n-f(n)-d-H(d)}$. This is possible since $n-f(n)-d-H(d) \leq n-|p|-d-H(d)$. □

Theorem 4.6. *Let α be a real which is weakly Chaitin random, and let V be an optimal computer. For every total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, if there exists an oracle deterministic Turing machine M such that, for infinitely many $n \in \mathbb{N}^+$, $M^{\operatorname{dom} V\restriction_n}(n) = \alpha\restriction_{n-f(n)}$, then the function f is not bounded to the above.* □

Using (5), Theorem 3.9 and Theorem 2.3, we can prove Theorem 4.6 as follows.

Proof of Theorem 4.6. Let α be a real which is weakly Chaitin random. Let V be an optimal computer, and let M be a deterministic Turing machine which computes V. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that there exists an oracle deterministic Turing machine M_0 such that, for infinitely many $n \in \mathbb{N}^+$, $M_0^{\operatorname{dom} V\restriction_n}(n) = \alpha\restriction_{n-f(n)}$. Note that, given (T_n^M, n) with $n \geq L_M$, one can calculate the finite set $\operatorname{dom} V\restriction_n$ by simulating the computation of M with the input p until at most T_n^M steps, for each $p \in \{0,1\}^*$ with $|p| \leq n$. Thus, we see that there exists a partial recursive function $\Psi\colon \mathbb{N} \times \mathbb{N}^+ \to \{0,1\}^*$ such that, for infinitely many $n \geq L_M$, $\Psi(T_n^M, n) = \alpha\restriction_{n-f(n)}$. It follows from (5) that $H(\alpha\restriction_{n-f(n)}) \leq H(T_n^M, n) + O(1)$ for infinitely many $n \geq L_M$. Thus, by Theorem 3.9 we see that there exists an infinite subset S of $\mathbb{N}^+$ such that

$$H(\alpha\restriction_{n-f(n)}) \leq n + O(1) \tag{10}$$

for all $n \in S$.

In the case where the function $n-f(n)$ of n is bounded to the above on S, there exists $c \in \mathbb{N}$ such that, for every $n \in S$, $n-c \leq f(n)$, and therefore the function f itself is not bounded to the above. Thus, in what follows, we assume that the function $n-f(n)$ of n is not bounded to the above on S. Thus we can choose a sequence $n_1, n_2, n_3, \ldots$ in S such that $1 \leq n_j - f(n_j) < n_{j+1} - f(n_{j+1})$ for all $j \in \mathbb{N}^+$. It follows from (10) that $H(\alpha\restriction_{n_j-f(n_j)}) - (n_j - f(n_j)) \leq f(n_j) + O(1)$ for all $j \in \mathbb{N}^+$. On the other hand, since α is weakly Chaitin random, it follows from Theorem 2.3 that $\lim_{n\to\infty} H(\alpha\restriction_n) - n = \infty$. This implies that $\lim_{j\to\infty} f(n_j) = \infty$. Hence, the function f is not bounded to the above. □

5 Elaboration of the Turing reduction $\operatorname{dom} U \leq_T \Omega_U$

Theorem 5.1 (main result III)**.** *Let V and W be optimal computers, and let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. Then the following two conditions are equivalent:*

(i) There exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_n\}}(n) = \operatorname{dom} W\restriction_{n+f(n)-c}$, where the finite subset $\operatorname{dom} W\restriction_{n+f(n)-c}$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format.

(ii) The function f is bounded to the above. □

In order to prove the implication (i) ⇒ (ii) of Theorem 5.1, we need Theorem 5.2 below. For the purpose of understanding the statement of Theorem 5.2, we concisely review some definitions and results of the theory of relative randomness. See e.g. [12, 6] for the detail of the theory.

An *oracle computer* is an oracle deterministic Turing machine M with the input and output alphabet $\{0,1\}$ such that, for every subset A of $\{0,1\}^*$, the domain of definition of M^A is a prefix-free set. For each oracle computer M, each subset A of $\{0,1\}^*$, and each $s \in \{0,1\}^*$, $H_M^A(s)$ is defined by $H_M^A(s) = \min\{\, |p| \mid p \in \{0,1\}^* \,\&\, M^A(p) = s \,\}$ (may be ∞). An oracle computer R is said to be *optimal* if for every oracle computer M there exists $d \in \mathbb{N}$ such that, for every subset A of $\{0,1\}^*$ and every $s \in \{0,1\}^*$,

$$H_R^A(s) \leq H_M^A(s) + d. \tag{11}$$

It is then easy to see that there exists an optimal oracle computer. For any $\alpha \in \mathbb{R}$, we say that α is 2-*random* if there exist an optimal oracle computer R and $c \in \mathbb{N}$ such that $n - c \leq H_R^{\mathrm{dom}\, U}(\alpha\restriction_n)$ for all $n \in \mathbb{N}^+$. Recall here that U is the optimal computer used to define $H(s)$.

For any $\alpha \in \mathbb{R}$, we say that α is *strongly Chaitin random* if there exists $c \in \mathbb{N}$ such that, for infinitely many $n \in \mathbb{N}^+$, $n + H(n) - c \leq H(\alpha\restriction_n)$. J. Miller recently showed the following theorem. See [6, 12] for the detail.

Theorem 5.2 (J. Miller)**.** *For every $\alpha \in \mathbb{R}$, α is strongly Chaitin random if and only if α is 2-random.* □

The implication (i) ⇒ (ii) of Theorem 5.1 is then proved as follows, based on Lemma 4.5, Theorem 5.2, and the fact that Ω_V is an r.e. real.

Proof of (i) ⇒ (ii) of Theorem 5.1. Let V and W be optimal computers. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that there exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_n\}}(n) = \mathrm{dom}\, W\restriction_{n+f(n)-c}$. Then, by considering the following procedure, we first see that $n + f(n) < H(\Omega_V\restriction_n) + O(1)$ for all $n \in \mathbb{N}^+$.

Given $\Omega_V\restriction_n$, one first calculates the finite set $\mathrm{dom}\, W\restriction_{n+f(n)-c}$ by simulating the computation of M with the input n and the oracle $\Omega_V\restriction_n$. Then, by calculating the set $\{\, W(p) \mid p \in \mathrm{dom}\, W\restriction_{n+f(n)-c}\}$ and picking any one finite binary string s which is not in this set, one can obtain $s \in \{0,1\}^*$ such that $n + f(n) - c < H_W(s)$.

Thus, there exists a partial recursive function $\Psi\colon \{0,1\}^* \to \{0,1\}^*$ such that, for all $n \in \mathbb{N}^+$, $n + f(n) - c < H_W(\Psi(\Omega_V\restriction_n))$. It follows from the optimality of W and (5) that

$$n + f(n) < H(\Omega_V\restriction_n) + O(1) \tag{12}$$

for all $n \in \mathbb{N}^+$.

Now, let us assume contrarily that the function f is not bounded to the above. Then it follows from Lemma 4.5 that $H(n) \leq f(n)$ for infinitely many $n \in \mathbb{N}^+$. Combining this with (12) we see that Ω_V is strongly Chaitin random. Thus, by Theorem 5.2, Ω_V is 2-random and therefore there exist an optimal oracle computer R and $d \in \mathbb{N}$ such that

$$n - d \leq H_R^{\mathrm{dom}\, U}(\Omega_V\restriction_n) \tag{13}$$

for all $n \in \mathbb{N}^+$.

On the other hand, $\Omega_V \leq_T \mathrm{dom}\, U$ holds, as shown in Theorem 3.1 in a stronger form. Thus, using (11) we can show that $H_R^{\mathrm{dom}\, U}(\Omega_V\restriction_n) \leq 2\log_2 n + O(1)$ for all $n \in \mathbb{N}^+$. However, this contradicts (13), and the proof is completed. □

On the other hand, in order to prove the implication (ii) $\Rightarrow$ (i) of Theorem 5.1, we need Theorem 5.3 below. Theorem 5.3 can be proved based on Fact 1 and Corollary 3.8. For completeness, however, we include in Appendix C a direct and self-contained proof of Theorem 5.3 without using Corollary 3.8.

Theorem 5.3. *Let V be an optimal computer, and let C be a computer. Then there exist an oracle deterministic Turing machine M and $d \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_{n+d}\}}(n) = \mathrm{dom}\, C\restriction_n$, where the finite subset $\mathrm{dom}\, C\restriction_n$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format.* □

Then the proof of the implication (ii) $\Rightarrow$ (i) of Theorem 5.1 is as follows.

Proof of (ii) $\Rightarrow$ (i) of Theorem 5.1. Let V and W be optimal computers. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that the function f is bounded to the above. Then there exists $d_1 \in \mathbb{N}$ such that $f(n) \leq d_1$ for all $n \in \mathbb{N}^+$. On the other hand, by Theorem 5.3, there exist an oracle deterministic Turing machine M and $d_2 \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_{n+d_2}\}}(n) = \mathrm{dom}\, W\restriction_n$, where the finite subset $\mathrm{dom}\, W\restriction_n$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format. We set $c = d_1 + d_2$. Then, by the following procedure, we see that there exists an oracle deterministic Turing machine M such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_n\}}(n) = \mathrm{dom}\, W\restriction_{n+f(n)-c}$.

Given n and $\Omega_V\restriction_n$ with $n > d_2$, one first calculates the finite set $\mathrm{dom}\, W\restriction_{n-d_2}$ by simulating the computation of M with the input $n - d_2$ and the oracle $\{\Omega_V\restriction_n\}$. One then calculates and outputs $\mathrm{dom}\, W\restriction_{n+f(n)-c}$. This is possible since $n + f(n) - c \leq n + d_1 - c = n - d_2$. □

Note that, as a variant of Theorem 5.1, we can prove the following theorem as well, in a similar manner to the proof of Theorem 5.1.

Theorem 5.4 (variant of the main result III)**.** *Let V and W be optimal computers, and let $f\colon \mathbb{N}^+ \to \mathbb{N}$ be a total recursive function. Then the following two conditions are equivalent:*

(i) There exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_{n-f(n)+c}\}}(n) = \mathrm{dom}\, W\restriction_n$, where the finite subset $\mathrm{dom}\, W\restriction_n$ of $\{0,1\}^*$ is represented as a finite binary string in a certain format.

(ii) The function f is bounded to the above. □

For completeness, we give a proof of the implication (i) $\Rightarrow$ (ii), i.e., the difficult part, of Theorem 5.4 as follows.

Proof of (i) $\Rightarrow$ (ii) of Theorem 5.4. Let V and W be optimal computers. For an arbitrary total recursive function $f\colon \mathbb{N}^+ \to \mathbb{N}$, assume that there exist an oracle deterministic Turing machine M and $c \in \mathbb{N}$ such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V\restriction_{n-f(n)+c}\}}(n) = \mathrm{dom}\, W\restriction_n$. Then, by considering the following procedure, we first see that $n < H(n, \Omega_V\restriction_{n-f(n)+c}) + O(1)$ for all $n \in \mathbb{N}^+$.

Given n and $\Omega_V\restriction_{n-f(n)+c}$, one first calculates the finite set $\mathrm{dom}\, W\restriction_n$ by simulating the computation of M with the input n and the oracle $\{\Omega_V\restriction_{n-f(n)+c}\}$. Then, by calculating the set $\{\, W(p) \mid p \in \mathrm{dom}\, W\restriction_n\}$ and picking any one finite binary string s which is not in this set, one can obtain $s \in \{0,1\}^*$ such that $n < H_W(s)$.

Thus, there exists a partial recursive function $\Psi\colon \mathbb{N}^+ \times \{0,1\}^* \to \{0,1\}^*$ such that, for all $n \in \mathbb{N}^+$, $n < H_W(\Psi(n, \Omega_V\restriction_{n-f(n)+c}))$. It follows from the optimality of W and (5) that

$$n < H(n, \Omega_V\restriction_{n-f(n)+c}) + O(1) \tag{14}$$

for all $n \in \mathbb{N}^+$.

Now, let us assume contrarily that the function f is not bounded to the above. It is then easy to show that there exists an increasing total recursive function $g\colon \mathbb{N}^+ \to \mathbb{N}^+$ such that the function $f(g(k))$ of k is increasing. Note that $H(s,t) \leq H(s) + H(t) + O(1)$ holds for all $s,t \in \{0,1\}^*$ by (3). It follows from (14) that

$$g(k) - H(g(k)) < H(\Omega_V\restriction_{g(k)-f(g(k))+c}) + O(1) \tag{15}$$

for all $k \in \mathbb{N}^+$. On the other hand, note that $\lim_{n\to\infty} n - H(n) = \infty$ holds by (4). Hence we have $\lim_{k\to\infty} g(k) - H(g(k)) = \infty$ since the function g is increasing. Based on (15), it is then easy to see that the function $g(k) - f(g(k)) + c$ of k is not bounded to the above. Therefore there exists an increasing total recursive function $h\colon \mathbb{N}^+ \to \mathbb{N}^+$ such that $g(h(l)) - f(g(h(l))) + c \geq 1$ for all $l \in \mathbb{N}^+$ and the function $g(h(l)) - f(g(h(l))) + c$ of l is increasing. For clarity, we define a total recursive function $m\colon \mathbb{N}^+ \to \mathbb{N}^+$ by $m(l) = g(h(l)) - f(g(h(l))) + c$. Since m is an increasing function, it is then easy to see that there exists a partial recursive function $\Phi\colon \mathbb{N}^+ \to \mathbb{N}^+$ such that $\Phi(m(l)) = g(h(l))$ for all $l \in \mathbb{N}^+$. Thus, based on (5), it is shown that

$$H(g(h(l)), \Omega_V\restriction_{m(l)}) \leq H(\Omega_V\restriction_{m(l)}) + O(1)$$

for all $l \in \mathbb{N}^+$. It follows from (14) that

$$m(l) + f(g(h(l))) < H(\Omega_V\restriction_{m(l)}) + O(1) \tag{16}$$

for all $l \in \mathbb{N}^+$. On the other hand, note that the total recursive function $f(g(h(l)))$ of l is increasing and therefore not bounded to the above. Thus, in a similar manner to the proof of Lemma 4.5 we can show that $H(m(l)) \leq f(g(h(l)))$ for infinitely many $l \in \mathbb{N}^+$. It follows from (16) that

$$m(l) + H(m(l)) < H(\Omega_V\restriction_{m(l)}) + O(1)$$

for infinitely many $l \in \mathbb{N}^+$. Since the function m is increasing, we see that Ω_V is strongly Chaitin random.

Hereafter, in the same manner as the proof of the implication (i) $\Rightarrow$ (ii) of Theorem 5.1, we can derive a contradiction using Theorem 5.2. This completes the proof. □

Acknowledgments

This work was supported by KAKENHI, Grant-in-Aid for Scientific Research (C) (20540134), by SCOPE from the Ministry of Internal Affairs and Communications of Japan, and by CREST from Japan Science and Technology Agency.

References

[1] C. S. Calude, P. H. Hertling, B. Khoussainov, and Y. Wang, "Recursively enumerable reals and Chaitin Ω numbers," *Theoret. Comput. Sci*, vol. 255, pp. 125–149, 2001.

[2] G. J. Chaitin, "A theory of program size formally identical to information theory," *J. Assoc. Comput. Mach.*, vol. 22, pp. 329–340, 1975.

[3] G. J. Chaitin, "Incompleteness theorems for random reals," *Adv. in Appl. Math.*, vol. 8, pp. 119–146, 1987.

[4] G. J. Chaitin, *Algorithmic Information Theory*. Cambridge University Press, Cambridge, 1987.

[5] G. J. Chaitin, "Program-size complexity computes the halting problem," *Bulletin of the European Association for Theoretical Computer Science*, vol. 57, p. 198, October 1995.

[6] R. G. Downey and D. R. Hirschfeldt, *Algorithmic Randomness and Complexity*. Springer-Verlag, To appear.

[7] P. Gács, "On the symmetry of algorithmic information," *Soviet Math. Dokl.*, vol. 15, pp. 1477–1480, 1974; correction, ibid. vol. 15, pp. 1480, 1974.

[8] A. Kučera and T. A. Slaman, "Randomness and recursive enumerability," *SIAM J. Comput.*, vol. 31, No. 1, pp. 199–211, 2001.

[9] L. A. Levin, "Laws of information conservation (non-growth) and aspects of the foundations of probability theory," *Problems of Inform. Transmission*, vol. 10, pp. 206–210, 1974.

[10] P. Martin-Löf, "The definition of random sequences," *Information and Control*, vol. 9, pp. 602–619, 1966.

[11] J. Miller and L. Yu, "On initial segment complexity and degrees of randomness," *Trans. Amer. Math. Soc.*, vol. 360, pp. 3193–3210, 2008.

[12] A. Nies, *Computability and Randomness*. Oxford University Press Inc., New York, 2009.

[13] C.-P. Schnorr, "Process complexity and effective random tests," *J. Comput. System Sci.*, vol. 7, pp. 376–388, 1973.

[14] R. M. Solovay, "Draft of a paper (or series of papers) on Chaitin's work ... done for the most part during the period of Sept.–Dec. 1974," unpublished manuscript, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, May 1975, 215 pp.

A The proof of Theorem 3.9

We here prove Theorem 3.9 using Lemma A.1 below. Let V be an optimal computer, and let M be a deterministic Turing machine which computes V.

Lemma A.1. *There exists $d \in \mathbb{N}$ such that, for every $p \in \operatorname{dom} V$, there exists $q \in \operatorname{dom} V$ for which $|q| \leq |p| + d$ and the running time of M on the input q is longer than the running time of M on the input p.*

Proof. Consider the computer C such that (i) $\mathrm{dom}\, C = \mathrm{dom}\, V$ and (ii) for every $p \in \mathrm{dom}\, V$, $C(p) = 1^{2|p|+T(p)+1}$, where $T(p)$ is the running time of M on the input p. It is easy to see that such a computer C exists. Then, since V is an optimal computer, from the definition of an optimal computer there exists $d_1 \in \mathbb{N}$ with the following property; if $p \in \mathrm{dom}\, C$, then there is q for which $V(q) = C(p)$ and $|q| \leq |p| + d_1$.

Thus, for each $p \in \mathrm{dom}\, V$ with $|p| \geq d_1$, there is q for which $V(q) = C(p)$ and $|q| \leq |p| + d_1$. It follows that

$$|V(q)| = 2\,|p| + T(p) + 1 > |p| + d_1 + T(p) \geq |q| + T(p). \tag{17}$$

Note that exactly $|q|$ cells on the tapes of M have the symbols 0 or 1 in the initial configuration of M with the input q, while at least $|V(q)|$ cells on the tape of M, on which the output is put, have the symbols 0 or 1 in the resulting final configuration of M. Since M can write at most one 0 or 1 on the tape, on which an output is put, every one step of its computation, the running time $T(q)$ of M on the input q is bounded to the below by the difference $|V(q)| - |q|$. Thus, by (17), we have $T(q) > T(p)$.

On the other hand, since $\mathrm{dom}\, V$ is not a recursive set, the function T_n^M of $n \geq L_M$ is not bounded to the above. Therefore, there exists $r_0 \in \mathrm{dom}\, V$ such that, for every $p \in \mathrm{dom}\, C$ with $|p| < d_1$, $T(r_0) > T(p)$. By setting $d_2 = |r_0|$ we then see that, for every $p \in \mathrm{dom}\, C$ with $|p| < d_1$, $|r_0| \leq |p| + d_2$.

Thus, by setting $d = \max\{d_1, d_2\}$ we see that, for every $p \in \mathrm{dom}\, V$, there is $q \in \mathrm{dom}\, V$ for which $|q| \leq |p| + d$ and $T(q) > T(p)$. This completes the proof. □

The proof of Theorem 3.9 is given as follows.

Proof of Theorem 3.9. By considering the following procedure, we first show that $n \leq H(T_n^M, n) + O(1)$ for all $n \geq L_M$.

Given (T_n^M, n) with $n \geq L_M$, one first calculates the finite set $\mathrm{dom}\, V\restriction_n$ by simulating the computation of M with the input p until at most T_n^M steps, for each $p \in \{0,1\}^*$ with $|p| \leq n$. Then, by calculating the set $\{\, V(p) \mid p \in \mathrm{dom}\, V\restriction_n \}$ and picking any one finite binary string s which is not in this set, one can obtain $s \in \{0,1\}^*$ such that $n < H_V(s)$.

Hence, there exists a partial recursive function $\Psi\colon \mathbb{N} \times \mathbb{N}^+ \to \{0,1\}^*$ such that, for all $n \geq L_M$, $n < H_V(\Psi(T_n^M, n))$. It follows from the optimality of V and (5) that $n < H(T_n^M, n) + O(1)$ for all $n \geq L_M$.

For each $p \in \mathrm{dom}\, V$, let $T(p)$ be the running time of M on the input p. It follows from Lemma A.1 that there exists $d \in \mathbb{N}$ such that, for every $p \in \mathrm{dom}\, V$, there exists $q \in \mathrm{dom}\, V$ for which $|q| \leq |p| + d$ and $T(q) > T(p)$. By considering the following procedure, we next show that $H(T_n^M, n) \leq H(T_n^M) + O(1)$ for all $n \geq L_M$.

Given T_n^M with $n \geq L_M$, one first simulates the computation of M with the input p until at most T_n^M steps, one by one for each element p in $\{0,1\}^*$, in the order defined on the ordered set $\{0,1\}^*$. Due to the definition of T_n^M, during the simulations one can eventually find the first element p_0 of $\{0,1\}^*$ such that $T(p_0) = T_n^M$. For this p_0, $|p_0| \leq n$ due to the definition of T_n^M, and there exists $q \in \mathrm{dom}\, V$ for which $|q| \leq |p_0| + d$ and $T(q) > T_n^M$. For this q, $|q| > n$ due to the definition of T_n^M again. Therefore $d \geq 1$ and $|p_0| \leq n < |p_0| + d$. Thus, there are still only d possibilities of n, so that one needs only $\lceil \log_2 d \rceil$ bits more in order to determine n.

Thus, there exists a partial recursive function $\Phi\colon \mathbb{N} \times \{0,1\}^* \to \mathbb{N} \times \mathbb{N}^+$ such that, for every $n \geq L_M$, there exists $s \in \{0,1\}^*$ with the properties that $|s| = \lceil \log_2 d \rceil$ and $\Phi(T_n^M, s) = (T_n^M, n)$. It

follows from (5) and (3) that $H(T^M_n, n) \leq H(T^M_n) + \max\{H(s) \mid s \in \{0,1\}^* \ \&\ |s| = \lceil \log_2 d \rceil\} + O(1)$ for all $n \geq L_M$.

Finally, we show that $H(T^M_n) \leq n + O(1)$ for all $n \geq L_M$. Let us consider the computer C such that (i) $\mathrm{dom}\, C = \mathrm{dom}\, V$ and (ii) for every $p \in \mathrm{dom}\, V$, $C(p) = T(p)$. Obviously, such a computer C exists. Then, by (3) we see that, for every $p \in \mathrm{dom}\, V$, $H(T(p)) \leq |p| + O(1)$. For each $n \geq L_M$, it follows from the definition of T^M_n that there exists $r \in \mathrm{dom}\, V$ such that $|r| \leq n$ and $T(r) = T^M_n$. Hence, $H(T^M_n) = H(T(r)) \leq |r| + O(1) \leq n + O(1)$. This completes the proof. □

B The proof of Lemma 4.5

Lemma 4.5 is proved as follows.

Proof of Lemma 4.5. Contrarily, assume that there exists $c \in \mathbb{N}^+$ such that, for every $n \geq c$, $f(n) < H(n)$. Then, since f is not bounded to the above, it is easy to see that there exists a total recursive function $\Psi\colon \mathbb{N}^+ \to \mathbb{N}^+$ such that, for every $k \in \mathbb{N}^+$, $k < H(\Psi(k))$. Thus, using (5) we see that $k < H(k) + O(1)$ for all $k \in \mathbb{N}^+$. On the other hand, using (4) we have $H(k) \leq 2\log_2 k + O(1)$ for all $k \in \mathbb{N}^+$. Therefore $k < 2\log_2 k + O(1)$ for all $k \in \mathbb{N}^+$. However, we have a contradiction on letting $k \to \infty$ in this inequality, and the result follows. □

C The proof of Theorem 5.3

In what follows, we prove Theorem 5.3 in a direct manner without using Corollary 3.8.

Proof of Theorem 5.3. In the case where $\mathrm{dom}\, C$ is a finite set, the result is obvious. Thus, in what follows, we assume that $\mathrm{dom}\, C$ is an infinite set.

Let $p_0, p_1, p_2, p_3, \ldots$ be a particular recursive enumeration of $\mathrm{dom}\, C$, and let D be a computer such that $\mathrm{dom}\, D = \mathrm{dom}\, C$ and $D(p_i) = i$ for all $i \in \mathbb{N}$. Recall here that we identify $\{0,1\}^*$ with $\mathbb{N}$. It is also easy to see that such a computer D exists. Since V is an optimal computer, from the definition of optimality of a computer there exists $d \in \mathbb{N}$ such that, for every $i \in \mathbb{N}$, there exists $q \in \{0,1\}^*$ for which $V(q) = i$ and $|q| \leq |p_i| + d$. Thus, $H_V(i) \leq |p_i| + d$ for every $i \in \mathbb{N}$. For each $s \in \{0,1\}^*$, we define $P_V(s)$ as $\sum_{V(p)=s} 2^{-|p|}$. Then, for each $i \in \mathbb{N}$,

$$P_V(i) \geq 2^{-H_V(i)} \geq 2^{-|p_i|-d}. \tag{18}$$

Then, by the following procedure, we see that there exists an oracle deterministic Turing machine M such that, for all $n \in \mathbb{N}^+$, $M^{\{\Omega_V \restriction_{n+d}\}}(n) = \mathrm{dom}\, C \restriction_n$.

Given n and $\Omega_V \restriction_{n+d}$, one can find $k_e \in \mathbb{N}$ such that $\sum_{i=0}^{k_e} P_V(i) > 0.(\Omega_V \restriction_{n+d})$. This is possible because $0.(\Omega_V \restriction_{n+d}) < \Omega_V$ and $\lim_{k\to\infty} \sum_{i=0}^{k} P_V(i) = \Omega_V$. It follows that

$$\sum_{i=k_e+1}^{\infty} P_V(i) = \Omega_V - \sum_{i=0}^{k_e} P_V(i) < \Omega_V - 0.(\Omega_V \restriction_{n+d}) < 2^{-n-d}.$$

Therefore, by (18),

$$\sum_{i=k_e+1}^{\infty} 2^{-|p_i|} \leq 2^d \sum_{i=k_e+1}^{\infty} P_V(i) < 2^{-n}.$$

It follows that, for every $i > k_e$, $2^{-|p_i|} < 2^{-n}$ and therefore $n < |p_i|$. Hence,

$$\operatorname{dom} C{\restriction_n} = \{\, p \in \operatorname{dom} C \mid |p| \leq n \,\} = \{\, p_i \mid i \leq k_e \;\&\; |p_i| \leq n \,\}.$$

Thus, by calculating the finite set $\{\, p_i \mid i \leq k_e \;\&\; |p_i| \leq n \,\}$, one can obtain the set $\operatorname{dom} C{\restriction_n}$. $\square$