On Characteristic Constants of Theories Defined by Kolmogorov Complexity

Shingo Ibuka, Makoto Kikuchi, and Hirotaka Kikyo

Dept. of Computer Science and Systems Engineering, Kobe University,
1-1 Rokkodai, Nada, Kobe 657-8501, Japan
ibuka@kurt.scitec.kobe-u.ac.jp, mkikuchi@kobe-u.ac.jp, kikyo@kobe-u.ac.jp

Abstract. Chaitin discovered that for each formal system $\mathbf{T}$, there exists a constant c such that no sentence of the form $K(x) > c$ is provable in $\mathbf{T}$, where $K(x)$ is the Kolmogorov complexity of x. We call the minimum such c the Chaitin characteristic constant of $\mathbf{T}$, or $c_{\mathbf{T}}$. There have been discussions about whether it represents the information content or strength of $\mathbf{T}$. Raatikainen tried to reveal the true source of $c_{\mathbf{T}}$, stating that it is determined by the smallest index of Turing machine which does not halt but we cannot prove this fact in $\mathbf{T}$. We call the index the Raatikainen characteristic constant of $\mathbf{T}$, denoted by $r_{\mathbf{T}}$. We show that $r_{\mathbf{T}}$ does not necessarily coincide with $c_{\mathbf{T}}$; for two arithmetical theories $\mathbf{T}$, $\mathbf{T}'$ with a Π_1-sentence provable in $\mathbf{T}'$ but not in $\mathbf{T}$, there is an enumeration of the Turing machines such that $r_{\mathbf{T}} < r_{\mathbf{T}'}$ and $c_{\mathbf{T}} = c_{\mathbf{T}'}$.

1 Introduction

Algorithmic information theory, originated by Andrey N. Kolmogorov, R. Solomonoff and Gregory J. Chaitin, brought a formalization of the information content of an individual object as Kolmogorov complexity. The Kolmogorov complexity of a number x is defined as the smallest code of Turing machine which outputs x. Chaitin[1] proved the incompleteness theorem in the following form: for a sound, finitely-specified, formal system $\mathbf{T}$ and an enumeration of the Turing machines, there exists a bound c such that $K(x) > c$ is not provable for any number x in $\mathbf{T}$. We call the minimum such c the Chaitin characteristic constant of $\mathbf{T}$, denoted by $c_{\mathbf{T}}$.

The received interpretation of this Chaitin's result is that the Chaitin characteristic constant $c_{\mathbf{T}}$ measures the information content or strength of the formal system $\mathbf{T}$. Michiel van Lambalgen[2] criticized the received interpretation and pointed out that $c_{\mathbf{T}}$ is not determined only by the theory $\mathbf{T}$, but also influenced by the choice of enumeration of the Turing machines. Panu Raatikainen[3] refined Lambalgen's argument and showed that for any formal system $\mathbf{T}$, there exists an enumeration of the Turing machines which makes $c_{\mathbf{T}}$ zero, or arbitrarily large. In the same paper he tried to give a characterization of $c_{\mathbf{T}}$. Let $r_{\mathbf{T}}$ denote the smallest code of Turing machine which does not halt but we cannot prove its non-halting property in $\mathbf{T}$. He stated that $c_{\mathbf{T}}$ is determined by $r_{\mathbf{T}}$. We call $r_{\mathbf{T}}$ *the Raatikainen characteristic constant of* $\mathbf{T}$.

W. Hodges and R. de Queiroz (Eds.): WoLLIC 2008, LNAI 5110, pp. 218–225, 2008.

The purpose of this paper is especially to show that $c_{\mathbf{T}}$ does not coincide with $r_{\mathbf{T}}$ contrary to Raatikainen's claim and to derive some mathematical properties of $c_{\mathbf{T}}$ and $r_{\mathbf{T}}$. First, we show that $r_{\mathbf{T}} \leq c_{\mathbf{T}}$ holds generally, but the converse not. By rearranging the enumeration of the Turing machines, we can make the difference between $r_{\mathbf{T}}$ and $c_{\mathbf{T}}$ arbitrarily large. Moreover, we prove that for two arithmetical theories $\mathbf{T}$, $\mathbf{T}'$ with a Π_1-sentence provable in $\mathbf{T}'$ but not in $\mathbf{T}$, there is an enumeration of the Turing machines such that $r_{\mathbf{T}} < r_{\mathbf{T}'}$ and $c_{\mathbf{T}} = c_{\mathbf{T}'}$.

Since these two characteristic constants are essentially what evaluate some information on Turing machines given by $\mathbf{T}$, the language of $\mathbf{T}$ is assumed to have some fixed way to express outputs of Turing machines. However, it is possible that a formal system with abundant axioms for outputs of Turing machines does not prove any facts on Kolmogorov complexity, because those two ideas may be expressed by different symbols which are not related by the axioms at all. For our purpose we consider formal systems defining Kolmogorov complexity naturally by Turing machines in this paper, while Raatikainen makes use of the recursion theorem and the soundness to dispense with this assumption. We confine our systems to first-order arithmetical theories, or formal systems in which first-order arithmetics can be interpreted, and assume that they are arithmetically sound, finitely-specified, and extending PA.

2 Arithmetizing Computability

Let PA denote the first-order Dedekind-Peano axioms, and $\mathcal{L}_A$ its language. Note that theories extending PA prove any Σ_1-sentence which is true in $\mathbb{N}$.

We consider Turing machines M of the following type. M has one semi-infinite tape, with the left end as the start cell. In one movement, it changes the state, writes a symbol on the tape, and makes the head move left or right, or stable. We assume the tape symbols are $0, 1$ and B, where B represents blank.

Definition 2.1. *Any ordered pair (x, y) of natural numbers can be coded by $\langle x, y\rangle = \frac{1}{2}(x + y)(x + y + 1) + x$.*

Tuples of natural numbers $(x_0, \ldots, x_n)$ can also be coded by a natural number $\langle x_0, x_1, \ldots, x_n\rangle$ defined inductively as follows for $n \geq 2$:

$$\langle x_0, x_1, \ldots, x_n\rangle = \langle x_0, \langle x_1, \ldots, x_n\rangle\rangle.$$

For $a = \langle x, y, n\rangle$, we define *Gödel β-function* as the function defined by

$$\beta(a, i) = x \bmod (y(i + 1) + 1) \text{ for each } i < n.$$

n is called the *length* of a, denoted by $|a|$. We also write $a[i]$ for $\beta(a, i)$.

Lemma 2.2. *Let $f : \mathbb{N} \to \mathbb{N}$ be any definable function in* PA. *Then*

$$\mathrm{PA} \vdash \forall n \exists a\, (\forall i < n \beta(a, i) = f(i) \wedge |a| = n).$$

For any $a_0, \ldots, a_{n-1} \in \mathbb{N}$, a natural number $a \in \mathbb{N}$ is called a *sequence number* for $a_0, \ldots, a_{n-1}$ if $\beta(a, i) = a_i$ for each natural number $i < n$.

Now we formulate the notion of Turing machine in PA.

Definition 2.3. *Suppose each cell in the tape is numbered 0, 1, 2, ... from the start cell to the right.*

(1) We code a transition rule of a Turing machines M by a tuple of the form

$$\langle q, s, q', s', m\rangle,$$

where $m \in \{L, R, S\}$. δ instructs that if the control is in state q over the cell with number s, it transitions into the state q' and writes s' on the cell, and moves or stay.

(2) A Turing machine M is coded by the sequence number

$$\langle N_Q, \delta, q_0, l_F\rangle,$$

where N_Q is the number of states of M, $q_0 < N_Q$ is the initial state of M, l_F is a sequence number coding the set of final states, and δ is a sequence number coding the set of transition rules.

(3) An instantaneous descriptions*, or an* ID*, is coded by the sequence number*

$$\langle q, t, h\rangle$$

such that $q < N_Q$ is a code of a state, t is a sequence number coding the contents of a contiguous finite sequence of cells covering every symbols appearing on the tape other than B, and h is the position of the head.

(4) A process *of M is coded by the sequence number of*

$$\langle ID_0, ID_1, \ldots, ID_l\rangle$$

of instantaneous descriptions such that the state of ID_0 is the initial state, and for each $i < l$, ID_{i+1} is obtained from ID_i by δ.

Next, we define some formulae describing movements of Turing machines.

There is a Δ_0-formula $\Psi_0(x, y)$ such that $\mathbb{N} \models \Psi_0(p, m)$ if and only if p is a number representing a process of the Turing machine coded by number m.

In order to describe the function represented by a Turing machine, we have to describe by a logical formula the relation between a natural number and the corresponding binary string representing it. The following are equivalent:

- $x = a_0 2^n + a_1 2^{n-1} + \ldots + a_{n-1} 2 + a_n$;
- There is a sequence x_0, x_1, ..., x_n such that $x_0 = a_0$, $x_i = 2x_{i-1} + a_i$ for $i = 1, \ldots, n$, and $x = x_n$.

Therefore, there is a Σ_1-formula $bin(x, t)$ which says that "t is a sequence number coding the binary representation of x."

So we have a Σ_1-formula $\Psi_1(m, y, z)$ which states that "z is an ID of the Turing machine coded by m and the contents of the tape is the binary representation of number y.

Using these formulae, we define a Σ_1-formula $\Psi_2(p, m, x)$ which states that "p is a number representing a valid sequence of ID of the Turing machine coded by m with the binary representation of x in the tape at the beginning.

Let φ_m the Turing machine coded by m and identify it with the computable (partial) function defined by φ_m. Then the statement "$\varphi_m(x) = y$" or "$\varphi_m(x) \downarrow y$" can be written by the formula

$$\begin{aligned}\exists p\,(\Psi_0(p,m) \wedge \exists n, t_0, t_2, i < p(n = |p| \wedge p[0] = \langle q_0, t_1, 0\rangle \\ \wedge p[n] = \langle q, t_2, 0\rangle \text{ for some } q \in F_m \\ \wedge bin(x, t_1) \wedge bin(y, t_2))\end{aligned}$$

Definition 2.4. *For two partial functions f and g on $\mathbb{N}$, $f \simeq g$ if and only if for any $x \in \mathbb{N}$,*

$$\begin{gathered}f(x) \text{ is defined} \iff g(x) \text{ is defined and} \\ f(x) = g(x) \text{ if both sides are defined.}\end{gathered}$$

We employ a fundamental result of recursion theory by Stephen C. Kleene.

Fact 2.5 (Kleene's recursion theorem). *If f is a total computable function, then there effectively exists a constant c such that $\varphi_{f(c)} \simeq \varphi_c$.*

The statement "Calculation of $\varphi_m(0)$ does not halt canonically", or "$\varphi_m(0)\uparrow$", can be represented by the formula

$$\forall p\,(\Psi_0(p,m) \wedge p[0] = \langle q_0, 0, 0\rangle \rightarrow p \text{ is not terminating canonically.})$$

Any effective enumeration of the computable functions (or recursive functions) can be represented by a universal Turing machine. Therefore, there is a computable total bijective function f (f is a "compiler" function) such that if m is a code ("program") of a function for given universal Turing machine then the partial function g coded by m with the given universal Turing machine satisfies $g \simeq \varphi_{f(m)}$.

Conversely, any computable bijective function f, $\varphi_{f(m)}$ $(m = 0, 1, \ldots)$ is an effective enumeration of any computable function.

We write φ^f_m for $\varphi_{f(m)}$. We also write $\varphi^f_m(x) \downarrow y$ if $\varphi_{f(m)}(x) \downarrow y$, and $\varphi^f_m(x) \uparrow$ if $\varphi_{f(m)}(x) \uparrow$. We will not specify the input if it is 0. Write $\varphi^f_m \uparrow$ for $\varphi^f_m(0) \uparrow$, $\varphi^f_m \downarrow y$ for $\varphi^f_m(0) \downarrow y$.

Note that any computable function is Σ_1-definable in PA.

Definition 2.6. *Let f be a Σ_1-definable bijective function in PA.*
Let $CT(d,m)$ be a formula saying that d is a code of ID representing a canonical termination of m, $tval(d,x)$ a formula saying that d is a code of ID with a tape value representing x.

We write $\varphi^f_m(x) \downarrow y$ for the formula

$$\exists p(\Psi_2(p, f(m), x) \wedge \exists l\,(|p| = l \wedge CT(p[l], m) \wedge tval(p[l], y))).$$

We write $\varphi^f_m \uparrow$ if the calculation of φ^f_m does not terminate canonically. This notion is equivalent to

$$\forall p(\Psi_2(p, f(m), x) \rightarrow \neg\exists l\,(|p| = l \wedge CT(p[l], m) \wedge tval(p[l], y))).$$

For a natural number n, the Kolmogorov complexity *of n with respect to f, denoted by $K^f(n)$, is the smallest natural number k such that $\varphi^f_k = n$. We can define $K^f(x) = y$ by "y is the smallest m such that $\exists p \Psi_2(p, f(m), 0, x)$."*

3 Characteristic Constants

Definition 3.1. *Let $\mathbf{T}$ be a formal system extending PA. We define two constants $\mathbf{c}_{f,\mathbf{T}}$ and $\mathbf{r}_{f,\mathbf{T}}$ as follows:*

$\mathbf{c}_{f,\mathbf{T}}$ is the smallest number k such that for any natural number n, $\mathbf{T} \nvdash K^f(n) > k$.

$\mathbf{r}_{f,\mathbf{T}}$ is the smallest number e such that $\varphi^f_e \uparrow$ and $\mathbf{T} \nvdash \varphi^f_e \uparrow$.

Theorem 3.2. *$c_{f,\mathbf{T}}$ and $r_{f,\mathbf{T}}$ exist for any sound, finitely-specified formal system $\mathbf{T}$ and a definable permutation f in* PA.

Proof. $c_{f,\mathbf{T}}$ exists, for there is a natural number c such that $\mathbf{T} \nvdash K^f(x) > c$. Define a total computable function i as follows. For each k, we have a Turing machine which searches the proof of $K^f(x) > k$ from $\mathbf{T}$ for some x and, if found, halts with output x. We can effectively obtain the index of this machine, $i(k)$. By recursion theorem, we can effectively find an index c such that $\varphi_c \simeq \varphi_{i(c)}$. If $\mathbf{T} \vdash K^f(x) > c$ for some natural number x, $\varphi_{i(c)} \simeq \varphi_c \downarrow x$, and hence $K^f(x) \leq c$. This contradicts the assumption of soundness of $\mathbf{T}$.

Next, we show that $r_{f,\mathbf{T}}$ exists. Consider a Turing machine $\varphi_{i(k)}$ which halts if $\mathbf{T} \vdash \varphi_k \uparrow$. By recursion theorem we have c such that $\varphi_{i(c)} \simeq \varphi_c$. φ_c does not halt, since if φ_c halts, φ_c does not halt by the soundness of $\mathbf{T}$. Hence, $\varphi_c \uparrow$. If $\mathbf{T} \vdash \varphi_c \uparrow$, then $\varphi_{i(c)}$, thus, φ_c halts. Contradiction. We have $\mathbf{T} \nvdash \varphi_c \uparrow$.

Next we argue the relation between $c_{f,\mathbf{T}}$ and $r_{f,\mathbf{T}}$. We first see $r_{f,\mathbf{T}} \leq c_{f,\mathbf{T}}$.

Lemma 3.3. *Let $\mathbf{T}$ be any formal system extending* PA*, and f any Σ_1-definable permutation, either $\mathbf{T} \vdash \varphi^f_i \uparrow$ or $\mathbf{T} \vdash \varphi^f_i \downarrow$ holds for any $i < r_{f,\mathbf{T}}$.*

Proof. Since $\mathbf{T}$ proves all Σ_1-sentences true in $\mathbb{N}$, if $\varphi^f_i \downarrow n, \mathbf{T} \vdash \varphi^f_i \downarrow n$. If $\varphi^f_i \uparrow$, by the minimality $r_{f,\mathbf{T}}$, $\mathbf{T} \vdash \varphi^f_i \uparrow$.

Theorem 3.4. *For any formal system $\mathbf{T}$ extending* PA *and any definable permutation f, $r_{f,\mathbf{T}} \leq c_{f,\mathbf{T}}$.*

Proof. Suppose $r_{f,\mathbf{T}} > c_{f,\mathbf{T}}$. Let $n \notin \{\varphi^f_0(0), \ldots, \varphi^f_{c_\mathbf{T}}(0)\}$ and $i \leq c_{f,\mathbf{T}}$. If $\mathbf{T} \vdash \varphi^f_i \downarrow k$, $\mathbf{T} \vdash \neg\varphi^f_i \downarrow n$. Otherwise, by lemma 3.3, $\mathbf{T} \vdash \varphi^f_i \uparrow$, $\mathbf{T} \vdash \neg\varphi^f_i \downarrow n$. Then $\mathbf{T} \vdash K^f(n) > c_{f,\mathbf{T}}$, a contradiction to the definition of $c_\mathbf{T}$.

Remark 3.5. For any definable permutation $\sigma : \mathbb{N} \to \mathbb{N}$ in PA, $PA \vdash \varphi^{f\circ\sigma}_m \simeq \varphi^f_{\sigma(m)}$.

However, $r_\mathbf{T} \geq c_\mathbf{T}$ does not necessarily hold. In fact, for a given formal system $\mathbf{T}$ we can enumerate the Turing machines so that $r_\mathbf{T} < c_\mathbf{T}$.

Lemma 3.6. *Let $f : \mathbb{N} \to \mathbb{N}$ be a bijective function Σ_1-definable in PA. There is a $g : \mathbb{N} \to \mathbb{N}$ Σ_1-definable in PA such that for any formal system* $\mathbf{T}$ *extending PA,*

(1) $\mathbf{T} \vdash \varphi^f_m \uparrow$ *if and only if* $\mathbf{T} \vdash \varphi^f_{g(m)} \uparrow$, *and*
(2) $\mathbf{T} \vdash \neg(\varphi^f_{g(m)} \downarrow 0)$.

Proof. We add transition rules to make the machine write 1 on the tape before termination. Let m be a number and $M = (Q, \Gamma, \delta, q_0, F)$ be the Turing machine coded by $f(m)$. Let $M' = (Q \cup \{q_f\}, \Gamma, \delta', q_0, \{q_f\})$ be a Turing machine such that q_f is a new state, and

$$\delta' = \delta \cup \{(q, b, q_f, 1, S) : b \in \{0, 1, B\}, q \in F\}.$$

Let m' be the code of M' and let $g(m) = f^{-1}(m')$.

We claim that g is the desired function.

First, we show (2), i.e., $PA \vdash \neg(\varphi^f_{g(m)} \downarrow 0)$. In case $\varphi^f_{g(m)} \uparrow$, then $\neg(\varphi^f_{g(m)} \downarrow 0)$. Otherwise, $\varphi^f_{g(m)} \downarrow y$ for some y. But $y \neq 0$ by the construction of M'.

Now, we show (1). Suppose $\mathbf{T} \vdash \varphi^f_m \uparrow$. Then $\mathbf{T} \vdash \forall p(\Psi_2(p, f(m), 0) \to \neg\exists l\,(|p| = l \wedge CT(p[l], m) \wedge tval(p[l], y)))$.

The following argument can be done in $\mathbf{T}$. Let p' be any natural number and assume that $\Psi_2(p', f(g(m)), x)$. Suppose that the last ID in p' has final state of M'. Truncate the last ID in p' and name it p. By the definition of M', p represents a valid calculation of "$f(m)$" which is canonically terminating. Therefore, we have $\varphi_{f(m)} \downarrow$, a contradiction. Hence, the last ID in p' does not have a final state of M'.

Conversely, suppose that

$$\mathbf{T} \vdash \forall p'(\Psi_2(p', m', 0) \to \neg\exists l\,(|p'| = l \wedge CT(p'[l], m') \wedge tval(p'[l], y))).$$

where $m' = f(g(m))$ is a code of the Turing machine M' described above.

Let p be any natural number and assume that $\Psi_2(p, f(m), x)$. If p represents a valid calculation of M terminating canonically, we can add an ID of M' to p to get a code p' of a valid calculation of M' terminating canonically. This contradicts with the assumption. Therefore, p does not represent a valid calculation of M terminating canonically.

Theorem 3.7. *Let* $\mathbf{T}$ *be any formal system extending PA. Let $f : \mathbb{N} \to \mathbb{N}$ be a bijective function Σ_1-definable in PA. Then there is a permutation σ on $\mathbb{N}$ definable in* PA *such that*

$$\mathbf{r}_{f,\mathbf{T}} = \mathbf{r}_{f\circ\sigma,\mathbf{T}} < \mathbf{c}_{f\circ\sigma,\mathbf{T}}.$$

Moreover, the difference between $\mathbf{r}_{f\circ\sigma,\mathbf{T}}$ and $\mathbf{c}_{f\circ\sigma,\mathbf{T}}$ can be arbitrarily large.

Proof. Let n be any natural number. We show that there is a permutation σ on $\mathbb{N}$ such that

$$\mathbf{r}_{f,\mathbf{T}} = \mathbf{r}_{f\circ\sigma,\mathbf{T}} < \mathbf{c}_{f\circ\sigma,\mathbf{T}}$$

and the difference of $\mathbf{r}_{f\circ\sigma,\mathbf{T}}$ and $\mathbf{c}_{f\circ\sigma,\mathbf{T}}$ is greater than n.

Let $g : \mathbb{N} \to \mathbb{N}$ be the function obtained in Lemma 3.6 with respect to f. Let σ be a permutation on $\mathbb{N}$ such that $\sigma(i) = g(i)$ for $i \leq \mathbf{r}_{f,\mathbf{T}} + n$. By Lemma 3.3, we have $\mathbf{T} \vdash \varphi_i^{f\circ\sigma} \downarrow$ or $\mathbf{T} \vdash \varphi_i^{f\circ\sigma} \uparrow$ for each $i < \mathbf{r}_{f,\mathbf{T}}$. But $\mathbf{T} \vdash \neg\varphi_i^{f\circ\sigma} \downarrow 0$ for $i \leq \mathbf{r}_{f,\mathbf{T}} + n$. Therefore, $\mathbf{T} \vdash K^{f\circ\sigma}(0) > \mathbf{r}_{f,\mathbf{T}} + n$. Hence, $\mathbf{r}_{f,\mathbf{T}} + n < \mathbf{c}_{f\circ\sigma,\mathbf{T}}$.

It is well-known that the truth definition of a Σ_n-formulas can be expressed by a Σ_{n+1}-formula. With a similar proof to this fact, we can show the following lemma:

Lemma 3.8. *Let $\psi(x)$ be a Δ_0-formula in $\mathcal{L}_A$. Then there is a code m_0 of a Turing machine such that*

$$PA \vdash \psi(x) \leftrightarrow \varphi_{m_0}(x) \downarrow 1,$$
$$PA \vdash \neg\psi(x) \leftrightarrow \varphi_{m_0}(x) \downarrow 0.$$

Lemma 3.9. *Let $\psi(x)$ be any Δ_0-formula in $\mathcal{L}_A$. Then there is a code m of a Turing machine such that*

$$PA \vdash \forall x\psi(x) \leftrightarrow \varphi_m \uparrow .$$

Proof. Let φ_{m_0} be a Turing machine we can get for $\psi(x)$ by Lemma 3.8.

Let φ_m be a Turing machine corresponding to the following C program:

```
while (ψ(x)) x++;
```

We explain φ_m more accurately. The initial state of φ_m is q_0. φ_m "saves" the value of x so that we can retrieve it later. Then φ_m evaluates $\psi(x)$ with the rules of φ_{m_0}. If the value is 0, then it enters the unique final state q_f and halts. If the value is 1, then it retrieves the value of x to an initial segment of the tape. Then it increments the value of x by 1 and enters the initial state q_0.

Now, we show the lemma. We are working in PA.

Suppose $\forall x\psi(x)$. We can show a formula in $\mathcal{L}_A$ expressing the following:

Claim 3.10 *For all n, there is a process of φ_m with input 0 such that the final ID has the initial state and the tape content is n.*

It is clear that the process with single ID $\langle q_0, bin(0), 0\rangle$ is the process for $n = 0$.

Assume $n \geq 1$. By induction hypothesis, there is a process of φ_m with input 0 such that the final ID is $\langle q_0, bin(n-1), 0\rangle$. Since $\forall x\psi(x)$, we have a process p_n for $\varphi_{m_0}(n-1) \downarrow 1$. We can concatenate process p_n and a process to perform $x++$ to the process already obtained. We have a process with the final ID $\langle q_0, bin(n), 0\rangle$. Therefore, we have the claim, and thus, $\varphi_m \uparrow$.

For the converse, suppose that $\varphi_m \uparrow$. We can prove a formula in $\mathcal{L}_A$ expressing the following claim:

Claim 3.11 *For all n, there is a process of φ_m with input 0 such that the final ID is $\langle q_0, bin(n), 0\rangle$, and if $n > 0$ then $\psi(n-1)$.*

We prove this by induction on n. It is obvious for $n = 0$.

Suppose $n \geq 1$. By the induction hypothesis, there is a process of φ_m with input 0 such that the final ID is $\langle q_0, bin(n-1), 0\rangle$.

By Lemma 3.8, there is a process for $\varphi_{m_0}(n-1) \downarrow 0$ or $\varphi_{m_0}(n-1) \downarrow 1$. In case $\varphi_{m_0}(n-1) \downarrow 0$, the control enters q_f and we have $\varphi_m \downarrow$. This contradicts our hypothesis. Therefore, there is a process for $\varphi_{m_0}(n-1) \downarrow 1$. Hence, we have $\psi(n-1)$ by Lemma 3.8. Now, we can increment the tape value x from $n-1$ to n. Therefore, we have the claim.

By the claim, we have $\forall x \psi(x)$.

By Lemma 3.9, we have the following theorem.

Theorem 3.12. *Let* $\mathbf{T}$ *and* $\mathbf{T}'$ *be sound, finitely-specified, formal systems extending* PA *such that* $\mathbf{T} < \mathbf{T}'$. *Then the following are equivalent:*

(1) There is a Π_1*-sentence* θ *such that* $\mathbf{T}' \vdash \theta$ *but* $\mathbf{T} \nvdash \theta$.
(2) There is an enumeration of the Turing machines such that $r_{\mathbf{T}} < r_{\mathbf{T}'}$.

Furthermore, we show that for two theories with a Π_1-gap, there is an enumeration of the Turing machines which makes them different in the Raatikainen constant but the same in the Chaitin characteristic constant.

Theorem 3.13. *Let* $\mathbf{T}, \mathbf{T}'$ *be sound, finitely-specified, formal systems extending* PA *with a* Π_1*-sentence provable in* $\mathbf{T}'$ *but not in* $\mathbf{T}$. *Then there exists an enumeration of the Turing machines such that* $c_{\mathbf{T}} = c_{\mathbf{T}'}$ *and* $r_{\mathbf{T}} < r_{\mathbf{T}'}$.

Proof. By Lemma 3.6 and Theorem 3.12, we can assume that $\mathbf{T} \nvdash \varphi_m \uparrow$, $\mathbf{T}' \vdash \varphi_m \uparrow$, and $\mathbf{T} \vdash \neg\varphi_m \downarrow 0$. By the same argument in Theorem 3.2, we take a c such that $\varphi_c \uparrow$ and $\mathbf{T}' \nvdash \neg\varphi_c \downarrow n$. Let f be a function such that $f(m) = 0$, $f(c) = 1$ and $f(x) = x$ otherwise. Since $\mathbf{T} \nvdash \varphi_0^f \uparrow$, $r_{\mathbf{T}}^f = 0$. By $\mathbf{T}' \vdash \varphi_0^f \uparrow$ and $\mathbf{T}' \nvdash \varphi_1^f \uparrow$, we have $r_{f,\mathbf{T}'} = 1$. Because $\mathbf{T} \vdash K^f(0) > 0$ and $T' \nvdash K^f(n) > 1$ for all n, $c_{f,\mathbf{T}} = c_{f,\mathbf{T}'} = 1$.

Remark 3.14. In the theorem above, we can make $c_{\mathbf{T}}$ arbitrarily large as in Theorem 3.7.

Acknowledgments. We wish to thank Norshasheema Shahidan for her valuable comments in discussions and on our draft.

References

1. Chaitin, G.J.: Information-theoretic limitations of formal systems. Journal of the ACM 21, 403–424 (1974)
2. van Lambalgen, M.: Algorithmic information theory. Journal of Symbolic Logic Logic 54, 1389–1400 (1989)
3. Raatikainen, P.: On interpreting Chaitin's incompleteness theorem. Journal of Philosophical Logic 27(6), 569–586 (1998)
4. Solovay, R.M.: A Version of Ω for Which ZFC Can Not Predict A Single Bit. CDMTCS-104 (1999)
5. Vitanyi, P., Li, M.: An Introduction to Kolmogorov Complexity and Its Applications, 2nd edn. Springer, Heidelberg (1997)