Vol.2012-AL-138 No.10
2012/1/28

On the Halting Problem of a Turing Machine

Eiichi Tanaka †1

In the series of reports the incompleteness theorems and the halting problem of a Turing machine were discussed on the general recursion theory. This report studies the latter on the set of partial recursive predicates, because a Turing machine can compute a partial recursive function. The formula $A_u(u)$ defined by the diagonal sequence $A_k(k)(k = 1, 2, 3, \cdots)$ of the set $\tilde{A}$ composed of all unary partial recursive formulas $A_k(u)(k = 1, 2, 3, \cdots)$ is essentially infinitely long. Based on this fact this paper shows that the predicate to express the computation of a Turing machine $(\exists z)T(x, y, z)$ is proved to be essentially infinitely long. This fact explains the undecidability of the halting problem of a Turing machine. In Appendix 1 the non-existence of the incompleteness theorems is stated based on the theory of computation.

1. Introduction

Gödel's incompleteness theorems [1] have been considered as one of the epoch making discoveries in the history of mathematics. Inspired by Gödel's paper, Turing [2] studied the halting problem of a computing machine that he proposed, namely, a Turing machine. He proved that the halting problem of a Turing machine is undecidable.

The halting problem is described as follows. Can we decide whether a computation of a given Turing machine with given any data exsists or not ? The problem was proved to be recursively undecidable. We shall study the length of the predicate for describing the halting problem .

A Turing machine can compute a partial recursive function. Therefore, we have to study the halting problem on the set of partial recursive functions (predicates). First of all, we shall prove that the predicate $A_u(u)$ defined by the diagonal sequence $A_k(k)(k = 1, 2, 3, \cdots)$ of the set $\tilde{A}$ composed of all unary partial recursive predicates $A_k(u)(k = 1, 2, 3, \cdots)$ is essentially infinitely long. Based on this fact this paper shows that the predicate proposed for proving the undecidability of the halting problem of a Turing machine is proved to be essentially infinitely long. In Appendix 1 we shall state the non-existence of the incompleteness theorems.

†1 Kobe University

2. Preliminaries

(1)Predicate logic.

The predicate logic for an arithmetic with addition and multiplication follows Shoenfield [8], but the classification of symbols is slightly modified.

Definition 1. The symbols of the predicate logic for the arithmetic are defined as follows. (a1) individual constants $(a, b, c, \cdots)$, (a2) variables $(x, y, z, \cdots)$, (a3) function symbols $(+, *)$, (a4) a predicate symbol $(=)$, (a5) logical symbols 1 $(\neg, \vee)$, (a6) a logical symbol 2 $(\exists)$, (a7) subsidiary symbols ((,), comma). $\exists$ is called an existential quantifier. In this paper let the basic symbols of the predicate logic for the arithmetic be the symbols of (a1) $\sim$ (a5) and (a7).

Let A and B be sets of collections of objects. A mapping from the set of n-tuples in A to B is called an n-ary function from A to B. A subset of the set of n-tuples in A is called an n-ary predicate in A. A partial mapping from A to B is a mapping from a subset of A to B. An n-ary partial function from A to B is a partial mapping from the set of n-tuples in A to B. An n-ary partial predicate in A is a partial subset of the set of n-tuples in A. An occurrence of variable x in predicate A is bound in A, if it occurs in a part of A of the form $\exists xA$, otherwise it is free in A.

Definition 2. (b1) An individual constant is a term.
(b2) A variable is a term.
(b3) If $t_1, t_2, \cdots, t_n$ are terms and f^n is an n-ary function, $f^n(t_1, t_2, \cdots, t_n)$ is a term.
(b4) Let $\{t_1, t_2, t_3, \cdots\}$ be an infinite set of terms. Define $T_1 = t_1 + t_2 + t_3 + \cdots$ and $T_2 = t_1 * t_2 * t_3 * \cdots$. T_1 and T_2 are terms.

Definition 3. (c1) If $t_1, t_2, \cdots, t_n$ are terms and P is an n-ary predicate,

$P(t_1, t_2, \cdots, t_n)$ is a formula.
(c2) If A and B are formulas, $A \vee B$ and $\neg A$ are formulas.
(c3) Let $\{A_1, A_2, A_3, \cdots\}$ be an infinite set of formulas. Define $F = A_1 \vee A_2 \vee A_3 \vee \cdots$. F is a formula.
(c4) If $C(x)$ is a formula and x is a free variable, $\exists x C(x)$ is a formula.

Formulas $(A \to B), (A \wedge B), (A \leftrightarrow B)$ and $\forall x A(x)$ are the abbreviations of $(\neg A \vee B), \neg(A \to \neg B), ((A \to B) \wedge (A \leftarrow B))$ and $\neg\exists\neg A(x)$, respectively. Symbol $\forall$ is a universal quantifier. A formula without free variables is called a closed formula or a sentence. We sometimes define function symbols and predicate symbols that are not in the basic symbols. In this paper we assume that defined function symbols and defined predicate symbols are rewritten using the basic symbols.

A proof is a finite sequence of one or more formulas such that each formula of the sequence is either an axiom or an immediate consequence of preceding formulas of the sequence. If A is the last formula in a proof P, P is said to be a proof of A. A is said to be provable or to be a theorem. Sometimes $\exists x C(x)$ represents finite numbers of Cs such as $C(x)(x = 1, 2, 3, \cdots)$. We call such an $\exists x C(x)$ a finite existential formula. If $\exists x C(x)$ represents an infinite number of Cs such as $C(x)(x = 1, 2, 3, \cdots)$, we call $\exists x C(x)$ an infinite existential formula.

Definition 4. The length of a function is defined as the number of the basic symbols in the function. If a function consists of infinitely many basic symbols, it is called an infinite length function. If a function is not an infinite length function, it is a finite length function. The lengths of a term, a formula and a predicate are similarly defined.

Terms T_1 and T_2 in Definition 2 are infinite length terms. Formula F in Definition 3 is an infinite length formula. Let Q be a finite length formula without infinite existential formulas. Assume that $\exists x C(x)$ is an infinite existential formula and there is an axiom or a theorem such that $\exists x C(x) \to Q$. $\exists x C(x)$ can be converted to a finite length formula.

Definition 5. If an infinite length function can not be transformed into a finite one in spite of every effort, the function is called an essentially infinite length function. An essentially infinite length formula and that predicate are defined in the similar way.

An expression is a sequence of symbols. If an expression consists of infinite symbols, it is called an infinite length expression. If not, it is a finite length expression. The concept of Gödel number is interesting and useful. However, there is a dicrepancy between the Gödel number of a predicate and its length. As we shall see later, the predicate $\exists z T(x, x, z)$ concerning the halting problem of a Turing machine is an example. That is, if we include $\exists$ in the basic symbol, the Gödel number of the predicate is finite, but its length may be essentially infinite. To remove this defect we must realize that the Gödel number of a predicate is finite if and only if its length is finite. This aim is attained by excluding a quantifier. This is why an existential quantifier is excluded from the basic symbols. Note that we need not expand a formula with a quantifier to one without it. As we shall see in Section 4, if a formula is a well formed non recursive formula, the length of it is infinitely long.

Many types of Gödel numbering have been proposed. We do not specify any particular numbering. Gödel numbering satisfies the following characteristics.
(*1) Different finite sequences of finite length expressions have different Gödel numbers.
(*2) There is an algorithm to decide whether a given number is the Gödel number of a finite sequence of finite length expressions or not. Therefore, we can reconstruct the finite sequence of finite length expressions from its Gödel number.

Consider the following two functions.

$$G_1(x) = S(x). \tag{1}$$

$$G_2(x) = g_1(x) * S(1) + g_2(x) * S(2) + g_3(x) * S(3) + \cdots. \tag{2}$$

where $g_k(x)$ is a function such that if $x = k$, $g_k(x) = 1$, and otherwise, $g_k(x) = 0$. $S(x)$ is an arbitrary function. $G_1(x)$ and $G_2(x)$ are the functions with the same

values, but their expressions are different. The Gödel number of $G_1(x)$ is finite and that of $G_2(x)$ is infinite. The Gödel number of a function is only based on its expression. $G_2(x)$ is not an essentially infinite length function, because that $G_2(x)$ can be rewritten to a finite length function $G_1(x)$.

The number of infinite length formulas is infinite, and the Gödel number of an infinite length formula is also infinite. Therefore, if a formula is an infinite length formula, we can not reconstruct the original formula from its Gödel number.

(*3) A finite sequence of finite length expressions satisfies (*1) and (*2).

(*4) If a finite sequence of expressions contains at least one infinite length expression, it does not satisfy (*1) and (*2).

Remark 1. Gödel numbering is effective for a finite sequence of finite length expressions. If a finite sequence of expressions contains at least one infinite length expression, Gödel numbering is not effective.

(2) Recursion theory

We shall describe a primitive recursive function (predicate), a recursive function (predicate) and a partial recursive function (predicate).

(d1) Initial functions

The initial functions are defined for natural numbers.

(a) The zero function: $Z(x) = 0$ for all x.

(b) The successor function: $S(x) = x'$ for all x, where x' is the successor of x.

(c) The projection functions: $U_n^i(\overline{x}) = x_i$ for $i = 1, 2, \cdots, n$.

(d2) Composition

Let $h_1, h_2, \cdots, h_r$ be r functions of n variables ($r \geq 1,\ n \geq 0$). Let g and f be a function of r variables and that of n variables, respectively. Define f as follows.

$$f(\overline{x}) = g(h_1(\overline{x}), \cdots, h_r(\overline{x})). \tag{3}$$

(d3) Primitive recursion

Let g and h be an n-ary ($n \geq 0$) function and an $(n+2)$-ary one, respectively. Define an $(n+1)$-ary function f as follows.

$$f(\overline{x}, 0) = g(\overline{x}). \tag{4}$$

$$f(\overline{x}, y') = h(\overline{x}, y, f(\overline{x}, y)). \tag{5}$$

Definition 6 A function is primitive recursive, if it can be obtained by a finite applications of (d2) and (d3) beginning with initial functions (d1). If P is an n-ary predicate, we define an n-ary function r_P such that $r_{P(a)} = 0$, if $P(a)$, and $r_{P(a)} = 1$, if $\neg P(a)$. We call r_P the representing function of P. P is primitive recursive, if r_P is primitive recursive.

(d4) μ operator

Function $g(\overline{x}, y)$ is called regular, if there is a natural number y such that $g(\overline{x}, y) = 0$ for any $\overline{x}$. μ operator is to find the least y for a regular function. Assume that g is an $(n+1)$-ary regular function. Define an n-ary function f as follows.

$$f(\overline{x}) = \mu y(g(\overline{x}, y) = 0). \tag{6}$$

Definition 7 A function is general recursive, if it can be obtained by finite applications of (d2) $\sim$ (d4) beginning with initial functions (d1). If the representing function of a predicate is general recursive, the predicate is general recursive.

If $f(\overline{x})$ and $g(\overline{x})$ have the same domain and the same value, we shall write

$$f(\overline{x}) = g(\overline{x}). \tag{7}$$

Assume that $h_k(\overline{x})(k = 1, \cdots, r)$ are defined and their values are $y_k(k = 1, \cdots, r)$. $g(h_1(\overline{x}), \cdots, h_r(\overline{x}))$ is defined, iff $g(y_1, \cdots, y_r)$ is defined. This is called the composition in the weak sense. Modify (d1) $\sim$ (d4) to define a partial recursive function.

(d1') Initial functions

The initial functions are partial functions.

(d2') Composition

The composition (d2) is done between partial functions in the weak sense.

(d3') Primitive recursion

The composition (5) is done between partial functions in the weak sense.

(d4') μ operator

The μ operator is defined under the condition that (6) is defined iff $g(\overline{x}, y)$ is defined and $g(\overline{x}, y) \neq 0$ for $y < y_0$.

Definition 8 A function is partial recursive, if it can be obtained by finite applications of (d2') $\sim$ (d4') beginning with initial functions (d1'). If the representing function of a predicate is partial recursive, the predicate is partial recursive.

Definition 9. A predicate $P(x)$ is recursively enumerable, if there is a recursive predicate $Q(x, y)$ such that

$$P(x) \leftrightarrow \exists y Q(x, y). \tag{8}$$

Definition 10. The symbols of the predicate logic for the theory of computation are defined as follows. Let (ek) be (ak) in Definition 1, where $k = 1, 2, 4, \cdots, 7$. Let (e3) be function symbols $(Z, S, U, +, *, \mu)$.

In this section the symbols of (e1) $\sim$ (e5) and (e7) are called the basic symbols for the theory of computation. Assume that defined function symbols and defined predicate symbols are rewritten using the basic symbols.

Definition 11. Let (fk) be (bk) in Definition 2, where $k = 1, 2, 3, 4$. The set of n-ary functions include the initial functions, the functions defined by composition and those by μ operator. (f5) If $t_1, t_2, \cdots, t_n$ are terms and $f^{(n+1)}(x_1, x_2, \cdots, x_n, y)$ is an $(n+1)$-ary function, $f^{(n+1)}(t_1, t_2, \cdots, t_n, y)$ is a term, where $f^{(n+1)}(x_1, x_2, \cdots, x_n, y)$ is defined by primitive recursion and $y = 0, 1, 2, \cdots$.

Definition 12. The terms generated by Definition 11 are called well formed terms (wfts, in abbreviation). The predicates generated by Definition 3 applying terms defined by Definition 11 are called well formed formulas (wffs, in abbreviation). Let W_t be the set of wfts and W_f be that of wffs. W_t includes infinite length terms and W_f does infinite length formulas.

3. Infinite Length Formulas

(1) Diagonal Sequences

a) General recursive predicates

Enumerate all finite length general recursive predicates with one free variable u.

$$A_1(u), A_2(u), A_3(u), \cdots \tag{9}$$

The set $\tilde{A}$ of finite length unary general recursive predicates is a countably infinite set. Consider predicates $I(u)$ and $J(u)$ such as

$$I(k) = \neg A_k(k) \quad (k = 1, 2, 3, \cdots). \tag{10}$$

$$J(k) = A_k(k) \quad (k = 1, 2, 3, \cdots). \tag{11}$$

(10) and (11) are called the antidiagonal sequence and the diagonal sequence of (9), respectively. If $I(u)$ is in $\tilde{A}$, it is a finite length general recursive predicate. Since $I(k) \neq A_k(k) (k = 1, 2, 3, \cdots)$, it is easy to prove by the diagonal method that $I(u)$ is not in $\tilde{A}$. Since $I(u)$ is not in $\tilde{A}$, it is not a finite length predicate. $I(u)$ can not be transformed to a finite length predicate. Therefore, it is an essentially infinite length predicate. Furthermore, we have

$$J(u) = \neg I(u). \tag{12}$$

Since $I(u)$ is an essentially infinite length predicate, so is $\neg I(u)$. That is, $J(u)$ is an essentially infinite length predicate.

Change the order of predicates in $\tilde{A}$. Let it be as follows.

$$A'_1(u), A'_2(u), A'_3(u), \cdots \tag{13}$$

Let $I'(u)$ and $J'(u)$ be the antidiagonal sequence and the diagonal sequence of (13), respectively. $I'(u)$ and $J'(u)$ are also essentially infinite length predicates. Note that there are infinitely many different sequences defined by all finite length unary general recursive predicates. For each sequence, there are an antidiagonal sequence and a diagonal sequence. Both of them are essentially infinite length sequences. If an antidiagonal sequence and a diagonal sequence are defined based on all finite length unary predicates, we need not pay attension to the order of predicates.

b) General recursive functions

Enumerate all finite length general recursive functions with one free variable u. Let them be $B_k(u)(k = 1, 2, 3, \cdots)$. Define a modified diagonal sequence $K(u)$ and the diagonal sequence $L(u)$ such as

$$K(k) = B_k(k) + 1 \quad (k = 1, 2, 3, \cdots). \tag{14}$$

$$L(k) = B_k(k) \quad (k = 1, 2, 3, \cdots). \tag{15}$$

It is easy to see that $K(u)$ is an essentially infinite length function. Note that

$$K(u) = L(u) + 1. \tag{16}$$

From (16) the diagonal sequence $L(u)$ is also an essentially infinite length function.

Lemma 1. Let $\tilde{A}$ be the set of all general recursive predicates with a free variable u and each member of $\tilde{A}$ be written as $A_k(u)(k = 1, 2, 3, \cdots)$. $A_u(u)$ is an essentially infinite length predicate. Let $\tilde{B}$ be the set of all general recursive functions with a free variable u and each member of $\tilde{B}$ be written as $B_k(u)(k = 1, 2, 3, \cdots)$. $B_u(u)$ is an essentially infinite length function.

c) Partial recursive predicates

Enumerate all finite length partial recursive predicates with a free variable u.

$$C_1(u), C_2(u), C_3(u), \cdots \tag{17}$$

The set $\tilde{C}$ of them is a countably infinite set. Let $C_k(h) = *$ indicate that predicate $C_k(h)$ is not defined for h. Define $\overline{C}_k(k)$=$true$ and $\neg\overline{C}_k(k)$=$false$ for $C_k(k) = *$, and $\overline{C}_k(k)$=$C_k(k)$ and $\neg\overline{C}_k(k)$=$\neg C_k(k)$ for $C_k(k) \neq *$. Consider predicates $M(u)$ and $N(u)$ such as

$$M(k) = \neg\overline{C}_k(k) \quad (k = 1, 2, 3, \cdots). \tag{18}$$

$$N(k) = \overline{C}_k(k) \quad (k = 1, 2, 3, \cdots). \tag{19}$$

If $M(u)$ is in $\tilde{C}$, it is a finite length partial recursive predicate. Since $M(k) \neq C_k(k)(k = 1, 2, 3, \cdots)$, it is easy to prove by the diagonal method that $M(u)$ is not in $\tilde{C}$. Since $M(u)$ is not in $\tilde{C}$, it is not a finite length predicate. $M(u)$ can not be transformed to a finite length predicate. Therefore, it is an essentially infinite length predicate. Furthermore, we have

$$N(u) = \neg M(u). \tag{20}$$

Since $M(u)$ is an essentially infinite length predicate, so is $N(u)$. That is, $N(u)$ is an essentially infinite length predicate. Consider the mapping

$$C_k(k) \to N(k)(k = 1, 2, 3, \cdots). \tag{21}$$

Note that the mapping is $* \to true$, $true \to true$ or $false \to false$. Furthermore the mapping is one to one from $C_u(u)$ to $N(u)$. That is, the mapping is a contruction mapping. $N(u)$ is an essentially infinite length predicate, so is the diagonal sequence $C_u(u)$.

d) Partial recursive functions

Enumerate all finite length partial recursive functions with a free variable u. Let them be $D_k(u)(k = 1, 2, 3, \cdots)$. Define $P(k)$ and $Q(k)$ as follows. If $D_k(k) = *$, $P(k) = 1$ and $Q(k) = 0$. If $D_k(k) \neq *$, $P(k) = D_k(k) + 1$ and $Q(k) = D_k(k)$. Note that

$$P(k) = Q(k) + 1. \tag{22}$$

It is easy to see that $P(u)$ and $Q(u)$ are essentially infinite length functions. Consider the mapping

$$D_k(k) \to N(k)(k = 1, 2, 3, \cdots), \tag{23}$$

Note that the mapping includes $* \to 0$ or $n \to n$, where n is a natural number. Furthermore the mapping is one to one from $D_u(u)$ to $Q(u)$. That is, the mapping is a contruction mapping. $Q(u)$ is an essentially infinite length predicate, so is the diagonal sequence $D_u(u)$.

Lemma 2. Let $\tilde{C}$ be the set of all partial recursive predicates with a variable u and each member of $\tilde{C}$ be written as $C_k(u)(k = 1, 2, 3, \cdots)$. $C_u(u)$ is an infinite length predicate. Let $\tilde{D}$ be the set of all partial recursive functions with a variable u and each member of $\tilde{D}$ be written as $D_k(u)(k = 1, 2, 3, \cdots)$. $D_u(u)$ is an infinite length function.

(2) The halting problem [6)]

A Turing machine is classified by a pair (m, n), where m and n are the number of symbols and that of states, respectively. m and n run on natural numbers. Let x, y and z be the Gödel number of a given Turing machine, a given data and the Gödel number of a Turing machine computation, respectively. Predicate $\exists z T(x, y, z)$ means that there is a computation z for a given Turing machine x

and a given data y, where $T(x, y, z)$ has been considered as a primitive recursive predicate. The halting problem is described as follows. Can we decide whether a computation of a given Turing machine with given any data exists or not ? The halting problem has been formulated as the decision problem of predicate $\exists z T(x, y, z)$. It is well known that predicate $\exists z T(x, x, z)$ is recursively undecidable.

Let t_k be the Gödel number of the Turing machine that computes a function $D_k(u)$. Then, $\exists z T(t_k, h, z)$ indicates the existence of the computation of $D_k(h)$. Therefore, the framework $T(x, y, z)$ can treat all unary finite length partial recursive functions. That is, $T(x, y, z)$ satisfies the premise of Lemma 2. Therefore $T(x, x, z)$ is an essentially infinite length predicate. So is $\exists z T(x, x, z)$.

Lemma 3. Predicate $T(x, x, z)$ is an essentially infinite length predicate.

We have proved that $T(x, x, z)$ is an essentially infinite length predicate. It goes without saying that $T(x, y, z)$, $\exists z T(x, x, z)$ and $\exists z T(x, y, z)$ are essentially infinite length predicates. That is, the undecidability of the halting problem is interpreted in the length of the predicate.

4. Concluding Remarks

We can summarize the conclusion just obtained as follows:
(1) A diagonal sequence and an antidiagonal sequence defined by all finite length unary partial recursive functions (predicates) are essentially infinite length unary functions (predicates).
(2) The predicate for the halting problem of a Turing machine is an essentially infinite length predicate. Therefore, it is very natural that the halting problem is undecidable.
(3) In Appendix 1 it is shown that there is no finite length predicate in W_f that can not be proved nor refuted. The arithmetic is consistent and the consistency of the arithmetic is provable in it. That is, the incompleteness theorems do not hold.

References

1) Gödel, K.: "Über Formal Unentscheidbare Sätze der Principia Mathematica und Verwandter Systeme, *Monatshefte für Mathematik und Physik*, Vol.38, pp.173-198 (1931).
2) Turing, T.: On Computable Numbers, with an Application to the Entscheidungsproblem, *Proceedings of the London Mathematical Society*, Ser.2, Vol.42, pp. 230-265 (1936).
3) Kleene, S. C.: Recursive Predicates and Quantifiers, *Transaction of the American Mathematical Society* , Vol.53, pp.41-74 (1936).
4) Kashima, R.: Incompleteness theories (in Japanese),*Lectures on Foundations of Mathematics - The Incompleteness Theorems and the Development (in Japanese)*, Nihonhyouronsha, Tokyo,(1997)
5) Tanaka, K.: *Logic of Arithmetic - Formal Systems and Non-standard Models - (in Japanese)*, Shokabo, Tokyo (2002).
6) Karp, C. R.: *Languages with expressions of infinite length* , North-Holland, Amsterdam (1964).
7) Davis, C.: *Computability and Unsolvability* , McGraw-Hill, NY (1958).
8) Shoenfield, J. R.: *Mathematical Logic*, Addison-Wesley, Massachusets (1967).
9) Heijenoort, J.: *From Frege to Gödel*, Harvard University Press, Massachusetts (1967).
10) Tanaka, E.: Reflections on Gödel and Turing, *Inf. Proc. Soc. Japan, SIG Technical Report* , Vol.2010-AL-131, No.12, pp.1-6 (2010).
11) Tanaka, E.: Reflections on the Diagonal Theorem, and related topics, *Inf. Proc. Soc. Japan, SIG Technical Report*, Vol.2011-AL-135, No.3, pp.1-8 (2009).

Appendix 1 From the Viewpoint of the Thory of Computation

We proved that $sub(x, x, z)$ is an essentially infinite length function [10]. This means that Gödel's proof of the incompleteness theorems is incorrect. Furthermore we revealed that the diagonal theorem does not hold [11]. The left problem is whether the incompleteness theorems exist or not. We shall discuss the problem and show the non-existence of the theorems. Hereafter, a recursive function (predicate) means a general recursive function (predicate).

(1) Recursive functions and predicates

We shall confirm that the computation of a primitive recursive function and that of a general recursive function terminate in finite operations.

Let $\overline{x}$ denote "$x_1, x_2, \cdots, x_n$" . The initial functions $Z(x) = 0$ and $S(x) = x'$ are determined, if x is given. $U_n^i(\overline{x}) = x_i$ is obtained, if $\overline{x}$ and i are given. If $\overline{x}, h_1, h_2, \cdots, h_r, g$ are given, a function $f(\overline{x})$ defined by composition can be computed.

Consider a primitive recursion. Assume that x and y are given. $f(x, y)$ is computed in the following way.

$$\begin{aligned} &f(x,0) = g(x).\\ &f(x,1) = h(x,0,f(x,0)) = h(x,0,g(x)).\\ &f(x,2) = h(x,1,f(x,1)).\\ &\quad\cdots\\ &f(x,y) = h(x,y-1,f(x,y-1)). \end{aligned} \tag{24}$$

The computation of a primitive recursion terminates in finite operations, if x and y are finite. A function obtained by finite applications of a composition and a primitive recursion beginning with initial functions is a function with finite symbols. Therefore a primitive recursive function is a finite length function. If r_P is primitive recursive, P is a finite length predicate.

Remark A-1. Both a primitive recursive function and a primitive recursive predicate are finite lengths.

A regular function $g(\overline{x}, y)$ has y_0 such that $g(\overline{x}, y_0) = 0$, where y_0 is a finite value. Even if y_0 is not known, y_0 can be obtained by finite times computations of $g(\overline{x}, y)$ for $y = 0, 1, 2, \cdots$. Define $p(z)$ such that if $z = 0, p(z) = 1$, and otherwise, $p(z) = 0$. The function $\mu g(\overline{x}, y_0)$ is expressed in the following way.

$$\mu g(\overline{x}, y_0) = 0 * p(g(\overline{x}, 0)) + 1 * p(g(\overline{x}, 1)) + \cdots + y_0 * p(g(\overline{x}, y_0)). \tag{25}$$

The function $\mu g(\overline{x}, y_0)$ is a finite length function. From (25), a recursive function is a finite length. If r_P is recursive, predicate P is a finite length.

Remark A-2. Both a recursive function and a recursive predicate are finite lengths.

(2) Well formed recursive functions and predicates

Recall the following important and widely accepted understanding of a recursive function.

(g1) A function is a computable function.

(g2) A computable function is a recursive function.

The latter is called Church's thesis. From Remark A-2 a recursive function is a finite length function. Note that we study a function in W_t. Therefore "a function" of "a finite length function" is "a recursive function". From Remark A-2, we have the following.

(g3) A recursive function is a finite length function in W_t, and vice versa.

From (g2) and (g3), we have the following.

(g4) A computable function is a finite length function in W_t, and vice versa.

We shall discuss predicates in W_f. Replacing a function with a predicate, we have the similar statements.

(h1) A recursive predicate is a decidable predicate.

(h2) A decidable predicate is a recursive predicate.

(h3) A recursive predicate is a finite length predicate in W_f, and vice versa.

(h4) A decidable predicate is a finite length predicate in W_f, and vice versa.

Note that since the general recursion theory includes Peano arithmetic, (h4) holds in the arithmetic. From (h4) we have the following lemma that denies the first incompleteness theorem.

Lemma A-1. There is no finite length predicate in W_f that can not be proved nor refuted.

Let A be any finite length recursive predicate. From (h4) A is decidable. $A \vee \neg A$ is tautology, provable, and decidable. Furthermore, $A \wedge \neg A$ is always false, unprovable and decidable. The arithmetic can not derive $A \wedge \neg A$. A theory T is inconsistent if every predicate of T is a theorem of T; otherwise, T is consistent. Therefore the arithmetic is consistent. The consistency of the arithmetic is provable in the arithmetic. The following lemma denies the second incompleteness theorem.

Lemma A-2. The arithmetic is consistent. The consistency of the arithmetic is provable in it.

(3) Well formed non recursive functions and predicates

Note that "a function" is "a well formed function". Since we study functions in W_t, the negation of "a finite length function" is "an essentially infinite length function in W_t", and the negation of "a recursive function" is "a non recursive function in W_t". The negation of both (g1) and (g2) introduces to (i1).
(i1) A non recursive function in W_t is a non computable function in W_t, and vice versa.

By the negation of both (g3) and (g4), we have (i2) and (i3), respectively.
(i2) A non recursive function in W_t is an essentially infinite length function in W_t, and vice versa..
(i3) A non computable function in W_t is an essentially infinite length function in W_t, and vice versa.

From (i2) we have the following.
(i4) A recursive enumerable but not recursive function is an essentially infinite length function.

Note that "a predicate" is "a well formed predicate". Since we study predicates in W_f, the negation of "a finite length predicate" is "an essentially infinite length predicate", and the negation of "a recursive predicate" is "a non recursive predicate in W_f". Similarly we have the followings.
(j1) A non recursive predicate in W_f is an undecidable predicate in W_f, and vice versa.
(j2) A non recursive predicate in W_f is an essentially infinite length predicate in W_f, and vice versa.
(j3) An undecidable predicate in W_f is an essentially infinite length predicate in W_f, and vice versa.

From (j2) we have the following.
(j4) A recursively enumerable but not recursive predicate is an essentially infinite length predicate.

Lemma A-3. A non recursive function in W_t is an essentially infinite length function. A non recursive predicate in W_f is an essentially infinite length predicate.

(4) Comments on some recursively enumerable predicates

(k1) Let r.e. indicate "recursively enumerable". Predicate $\exists z T(x,x,z)$ has been considered as r.e. but not recursive one [7]. However, $T(x,x,z)$ is an infinite length predicate, that is, the predicate is not recursive. Therefore $\exists z T(x,x,z)$ is not r.e.. The predicate is in W_f and not recursive. Then it is an infinite length predicate.
(k2) In the proof of Rice's theorem K appears, where $K = \{x | \exists z T(x,x,z)\}$ and K is supposed to be r.e.. Even if the wording is incorrect, Rice's theorem holds.
(k3) A non recursive function in W_f is an essentially infinite length function. A non recursive predicate in W_f is also an essentially infinite length predicate.
(k4) Gödel [1] states that $Bew_k(x)$ is not recursive. The predicate is in W_f. From Lemma A-3 it is an essentially infinite length predicate.

Appendix 2

Reflections on the Diagonal Theorem, and related topics, [11]

"Errata: Reflections on the Diagonal Theorem, and related topics" was distributed at the meeting (May 16,2011). We shall make it more concise.
p.1, left, ↓ line 7. "formulas" → "formula".
p.4, right, ↑ line 5. "(32)" → "(10)".
p.5, left ∼ p.6, left. Delete subsection "Peano arithmetic - Π_1 incompleteness".
p.7, right, ↓ line 6. "ser.2,42,pp.230-265,(1936)" → "pp.821-865".