

# カウンタ機械の計算可能性

神戸大学工学部情報知能工学科  
田村直之 (tamura@kobe-u.ac.jp)

2000年4月  
(修正 2002年5月)

## 1 カウンタ機械

カウンタ機械は以下から構成される

- カウンタ:  $x_0, x_1, \dots$   
可算無限個ある。各  $x_i$  は任意の自然数 (0 以上の整数) を記憶する。初期値は 0 である。
- プログラム: 有限列  $I_0, I_1, \dots, I_K$   
各  $I_i$  は、以下に示す基本命令である。
- プログラムカウンタ:  $PC$   
0 以上の自然数を記憶する。 $PC$  は、次に実行すべき命令の場所を示す。
- 入力装置, 出力装置:  
文字は 0, 1, スペースの 3 種類で、それぞれの文字コードは 0, 1, 2 である。

プログラムが与えられた時、実際に使用するカウンタは有限個であることに注意。  
基本命令

- $x_i := x_i + 1$
- $x_i := x_i - 1$
- if  $x_i \neq 0$  goto  $L$
- read  $x_0$
- write  $x_0$

動作

- プログラムカウンタ  $PC$  に 0 をセットする。
- $PC \leq K$  の間、以下のように基本命令を繰り返し実行する。
  - $I_{PC}$  を  $I$  とし、 $PC$  をインクリメントする。
  - $I$  が  $x_i := x_i + 1$  の時、 $x_i$  をインクリメントする。
  - $I$  が  $x_i := x_i - 1$  の時、 $x_i$  が正ならデクリメントする。
  - $I$  が if  $x_i \neq 0$  goto  $L$  の時、 $x_i \neq 0$  なら  $PC$  に  $L$  をセットする。
  - $I$  が read  $x_0$  の時、入力装置から読み込んだ文字コードを  $x_0$  をセットする。入力が終了すれば 3 を  $x_0$  にセットする。

- $I$  が write  $x_0$  の時,  $x_0$  が 0 なら文字 0 を, 1 なら文字 1 を, 2 以上ならスペースを出力装置に出す.

- $PC > K$  になれば, 停止する.

## 1.1 準備

$x_i$  に 0 をセットする命令列  $x_i := 0$  は以下のようにマクロ定義できる. ただし  $L_1$  はマクロ中の局所的なラベルである (以下のマクロ定義でも同様).

- $x_i := 0 \quad \equiv$   
 $L_1: x_i := x_i - 1$   
 if  $x_i \neq 0$  goto  $L_1$

$x_i$  に定数  $c$  をセットする命令列  $x_i := c$  は, 各々の  $c$  について以下のようにマクロ定義できる.

- $x_i := c \quad \equiv$   
 $x_i := 0$   
 $x_i := x_i + 1$  ( $c$  回繰り返して並べる)

分岐マクロは, 以下のように定義できる. ただし,  $x_t$  はプログラムの他の場所で利用していないカウンタである (以下のマクロ定義でも同様とし,  $x_t$  以外に  $x_u, x_v$  など用いる).

- goto  $L \quad \equiv$   
 $x_t := x_t + 1$   
 if  $x_t \neq 0$  goto  $L$
- if  $x_i = 0$  goto  $L \quad \equiv$   
 if  $x_i \neq 0$  goto  $L_1$   
 goto  $L$   
 $L_1:$

## 1.2 演算

代入を行うマクロ  $x_i := x_j$  は, 以下のように定義できる.

- $x_i := x_j \quad \equiv$   
 $x_i := 0$   
 $x_t := 0$   
 $L_1: \text{if } x_j = 0 \text{ goto } L_2$   
 $x_j := x_j - 1; x_i := x_i + 1; x_t := x_t + 1$   
 goto  $L_1$   
 $L_2: \text{if } x_t = 0 \text{ goto } L_3$   
 $x_t := x_t - 1; x_j := x_j + 1$   
 goto  $L_2$   
 $L_3:$

加減算を行うマクロは, それぞれ以下のように定義できる. ただし減算で  $x_i < x_j$  の時は, 0 がセットされる.

- $x_i := x_i + x_j \equiv$   
 $x_t := 0$   
 $L_1 : \text{ if } x_j = 0 \text{ goto } L_2$   
 $x_j := x_j - 1; x_t := x_t + 1; x_i := x_i + 1$   
 $\text{goto } L_1$   
 $L_2 : \text{ if } x_t = 0 \text{ goto } L_3$   
 $x_t := x_t - 1; x_j := x_j + 1$   
 $\text{goto } L_2$   
 $L_3 :$
- $x_i := x_i - x_j \equiv$   
 $x_t := 0$   
 $L_1 : \text{ if } x_j = 0 \text{ goto } L_2$   
 $x_j := x_j - 1; x_t := x_t + 1; x_i := x_i - 1$   
 $\text{goto } L_1$   
 $L_2 : \text{ if } x_t = 0 \text{ goto } L_3$   
 $x_t := x_t - 1; x_j := x_j + 1$   
 $\text{goto } L_2$   
 $L_3 :$

比較分岐マクロは、以下のように定義できる。

- $\text{if } x_i \geq x_j \text{ goto } L \equiv$   
 $x_t := 0$   
 $L_1 : \text{ if } x_j = 0 \text{ goto } L_2$   
 $\text{if } x_i = 0 \text{ goto } L_3$   
 $x_i := x_i - 1; x_j := x_j - 1; x_t := x_t + 1$   
 $\text{goto } L_1$   
 $L_2 : \text{ if } x_t = 0 \text{ goto } L$   
 $x_t := x_t - 1; x_i := x_i + 1; x_j := x_j + 1$   
 $\text{goto } L_2$   
 $L_3 : \text{ if } x_t = 0 \text{ goto } L_4$   
 $x_t := x_t - 1; x_i := x_i + 1; x_j := x_j + 1$   
 $\text{goto } L_3$   
 $L_4 :$
- $\text{if } x_i < x_j \text{ goto } L \equiv$   
 $\text{if } x_i \geq x_j \text{ goto } L_1$   
 $\text{goto } L$   
 $L_1 :$
- $\text{if } x_i = x_j \text{ goto } L \equiv$   
 $\text{if } x_i < x_j \text{ goto } L_1$   
 $\text{if } x_j < x_i \text{ goto } L_1$   
 $\text{goto } L$   
 $L_1 :$

- if  $x_i \neq x_j$  goto  $L$   $\equiv$   
     if  $x_i = x_j$  goto  $L_1$   
     goto  $L$   
    $L_1 :$

乗算を行うマクロは、以下のように定義できる。

- $x_i := x_i \times x_j$   $\equiv$   
      $x_t := x_i; x_i := 0$   
    $L_1 :$  if  $x_t = 0$  goto  $L_2$   
          $x_t := x_t - 1; x_i := x_i + x_j$   
         goto  $L_1$   
    $L_2 :$

除算 (切捨て) を行うマクロは、以下のように定義できる。除数が 0 の場合は、無限ループになる。

- $x_i := x_i \text{ div } x_j$   $\equiv$   
      $x_t := x_i; x_i := 0$   
    $L_1 :$  if  $x_t < x_j$  goto  $L_2$   
          $x_t := x_t - x_j; x_i := x_i + 1$   
         goto  $L_1$   
    $L_2 :$

割り切れるかどうかを判定するマクロは、以下のように定義できる。除数が 0 の場合は、無限ループになる。

- if  $x_i \bmod x_j = 0$  goto  $L$   $\equiv$   
      $x_t := x_i$   
    $L_1 :$  if  $x_t < x_j$  goto  $L_2$   
          $x_t := x_t - x_j$   
         goto  $L_1$   
    $L_2 :$  if  $x_t = 0$  goto  $L$
- if  $x_i \bmod x_j \neq 0$  goto  $L$   $\equiv$   
     if  $x_i \bmod x_j = 0$  goto  $L_1$   
     goto  $L$   
    $L_1 :$

以上の演算で、 $x_j$  が定数の場合も容易に定義できる。

$x_i$  が素数かどうかを判定するマクロは、以下のように定義できる。

- if isprime( $x_i$ ) goto  $L$   $\equiv$   
      $x_t := 2$   
     if  $x_i < x_t$  goto  $L_2$   
    $L_1 :$  if  $x_i = x_t$  goto  $L$   
         if  $x_i \bmod x_t = 0$  goto  $L_2$   
          $x_t := x_t + 1$   
         goto  $L_1$   
    $L_2 :$

$x_j$  番目の素数  $p_{x_j}$  を求めるマクロは、以下のように定義できる。ただし、 $p_0 = 2$  とする。

- $x_i := \text{prime}(x_j) \quad \equiv$   
 $x_i := 2; x_t := x_j$   
 $L_1 : \text{ if } x_t = 0 \text{ goto } L_3$   
 $x_i := x_i + 1; x_t := x_t - 1$
- $L_2 : \text{ if isprime}(x_i) \text{ goto } L_1$   
 $x_i := x_i + 1$   
 $\text{goto } L_2$
- $L_3 :$

以上の各マクロで、 $x_j$  を定数  $c$  に置き換えたマクロは以下のように定義すればよい ( $x_i := x_i + c$  の場合)。

- $x_i := x_i + c \quad \equiv$   
 $x_t := c$   
 $x_i := x_i + x_t$

### 1.3 配列

自然数の列  $z_0, z_1, \dots, z_m$  を一つのカウンタ  $x_i$  で以下のように表現する。

$$x_i = 2^{z_0} 3^{z_1} 5^{z_2} \dots p_m^{z_m} = \prod_{i=0}^m p_i^{z_i}$$

この時、 $x_i$  を配列と考える。配列を操作するためのマクロは以下のように定義できる。

- $\text{clear } x_i \quad \equiv$   
 $x_i := 1$
- $x_i[x_j] := x_i[x_j] + 1 \quad \equiv$   
 $x_t := \text{prime}(x_j)$   
 $x_i := x_i \times x_t$
- $x_i[x_j] := x_i[x_j] - 1 \quad \equiv$   
 $x_t := \text{prime}(x_j)$   
 $\text{if } x_i \bmod x_t \neq 0 \text{ goto } L_1$   
 $x_i := x_i \text{ div } x_t$   
 $L_1 :$
- $\text{if } x_i[x_j] \neq 0 \text{ goto } L \quad \equiv$   
 $x_t := \text{prime}(x_j)$   
 $\text{if } x_i \bmod x_t \neq 0 \text{ goto } L$

$x_i[x_j] := x_k, x_k := x_i[x_j]$  などが定義できることは明らか。

また、全入力を配列  $x_i$  に読み込むマクロは以下のように定義できる。

- $\text{Read\_Input } x_i \quad \equiv$   
 $\text{clear } x_i; x_t := 0$   
 $L_1 : \text{ read } x_0$   
 $x_i[x_t] := x_0; x_t := x_t + 1$   
 $\text{if } x_0 \neq 3 \text{ goto } L_1$

このような配列は、文字列を表している配列と考えることができる (終端は文字コード 3)。文字配列  $x_i$  と  $x_j$  を連結するマクロは以下のように定義できる。

- $x_i := x_i \# x_j \equiv$   
 $x_u := 0$   
 $L_1: \text{ if } x_i[x_u] = 3 \text{ goto } L_2$   
 $x_u := x_u + 1; \text{ goto } L_1$   
 $L_2: x_v := 0$   
 $L_3: x_t := x_j[x_v]; x_i[x_u] := x_t$   
 $x_u := x_u + 1; x_v := x_v + 1$   
 $\text{ if } x_t \neq 3 \text{ goto } L_3$

また、文字配列を表す定数をダブルクォートでくくって表す。

$$"010" = 2^0 3^{15} 2^7 0 11^3$$

## 2 2-カウンタ機械

カウンタ機械は、カウンタが  $a, b$  の二つだけの 2-カウンタ機械と同等である。2-カウンタ機械の基本命令は以下の通り。

$$\begin{aligned} a := a + 1, \quad a := a - 1, \quad \text{if } a \neq 0 \text{ goto } L, \quad \text{read } a, \quad \text{write } a \\ b := b + 1, \quad b := b - 1, \quad \text{if } b \neq 0 \text{ goto } L, \end{aligned}$$

カウンタ機械のカウンタ  $x_0, x_1, \dots, x_m$  を、以下のようにしてカウンタ  $b$  一つで表す。なお、 $b$  の初期値は 1 とする。

$$b = 2^{x_0} 3^{x_1} 5^{x_2} \dots p_m^{x_m} = \prod_{i=0}^m p_i^{x_i}$$

ここで  $p_i$  は  $i$  番目の素数である (2 を  $p_0$  とする)。

まず、2-カウンタ機械について、いくつかのマクロ命令を定義する。ただし、定数  $k$  は正の自然数とする。

- $a := 0 \equiv$   
 $L_1: a := a - 1$   
 $\text{ if } a \neq 0 \text{ goto } L_1$
- $\text{if } b > 0 \text{ goto } L \equiv$   
 $\text{ if } b \neq 0 \text{ goto } L$
- $\text{if } b > k \text{ goto } L \equiv$   
 $\text{ if } b \neq 0 \text{ goto } L_1$   
 $b := b + 1$   
 $\text{ if } b \neq 0 \text{ goto } L_3$   
 $L_1: b := b - 1$   
 $\text{ if } b > (k - 1) \text{ goto } L_2$   
 $b := b + 1$   
 $\text{ if } b \neq 0 \text{ goto } L_4$   
 $L_2: b := b + 1$   
 $\text{ if } b \neq 0 \text{ goto } L$   
 $L_3: b := b - 1$   
 $L_4:$

- $b := b \times k \equiv$   
 $a := 0; b := b + 1$   
 $L_1: \quad b := b - 1$   
 $\quad a := a + 1$  ( $k$  回繰り返して並べる)  
 $\quad \text{if } b \neq 0 \text{ goto } L_1$   
 $L_2: \quad a := a - 1; b := b + 1$   
 $\quad \text{if } a \neq 0 \text{ goto } L_2$   
 $\quad b := b - 1$  ( $k$  回繰り返して並べる)
- $b := b \text{ div } k \equiv$   
 $a := 0; b := b + 1$   
 $L_1: \quad b := b - 1$  ( $k$  回繰り返して並べる)  
 $\quad a := a + 1$   
 $\quad \text{if } b \neq 0 \text{ goto } L_1$   
 $L_2: \quad a := a - 1; b := b + 1$   
 $\quad \text{if } a \neq 0 \text{ goto } L_2$   
 $\quad b := b - 1$
- $\text{if } b \bmod k \neq 0 \text{ goto } L \equiv$   
 $a := 0; a := a + 1$   
 $L_1: \quad \text{if } b > (k - 1) \text{ goto } L_2$   
 $\quad \text{if } b \neq 0 \text{ goto } L_3$   
 $\quad \text{if } a \neq 0 \text{ goto } L_4$   
 $L_2: \quad b := b - 1; a := a + 1$  ( $k$  回繰り返して並べる)  
 $\quad \text{if } a \neq 0 \text{ goto } L_1$   
 $L_3: \quad a := a - 1; b := b + 1$   
 $\quad \text{if } a \neq 0 \text{ goto } L_3$   
 $\quad b := b - 1$   
 $\quad a := a + 1; \text{ if } a \neq 0 \text{ goto } L$   
 $L_4: \quad a := a - 1; b := b + 1$   
 $\quad \text{if } a \neq 0 \text{ goto } L_4$   
 $\quad b := b - 1$

カウンタ機械の基本命令は以下のようにマクロ定義できる。常に  $b \neq 0$  であることに注意。

- $x_i := x_i + 1 \equiv$   
 $b := b \times p_i$
- $x_i := x_i - 1 \equiv$   
 $\text{if } b \bmod p_i \neq 0 \text{ goto } L_1$   
 $\quad b := b \text{ div } p_i$   
 $L_1:$
- $\text{if } x_i \neq 0 \text{ goto } L \equiv$   
 $\text{if } b \bmod p_i \neq 0 \text{ goto } L_1$   
 $\text{if } b \neq 0 \text{ goto } L$   
 $L_1:$
- $\text{read } x_0 \equiv$

```

L1 : x0 := x0 - 1
      if x0 ≠ 0 goto L1
      read a
      if a ≠ 0 goto L2
      if b ≠ 0 goto L5
L2 : a := a - 1; if a ≠ 0 goto L3
      x0 := x0 + 1; if b ≠ 0 goto L5
L3 : a := a - 1; if a ≠ 0 goto L4
      x0 := x0 + 1; x0 := x0 + 1; if b ≠ 0 goto L5
L4 : x0 := x0 + 1; x0 := x0 + 1; x0 := x0 + 1
L5 :
• write x0 ≡
  if x0 ≠ 0 goto L1
  a := 0; write a; if b ≠ 0 goto L4
L1 : x0 := x0 - 1; if x0 ≠ 0 goto L2
      a := 0; a := a + 1; write a; if b ≠ 0 goto L3
L2 : a := 0; a := a + 1; a := a + 1; write a
L3 : x0 := x0 + 1
L4 :

```

### 3 万能 2-カウンタ機械

2-カウンタ機械のインタプリタをカウンタ機械で構成する。プログラムは配列  $x_1$  に格納し、プログラムサイズは変数  $x_2$  で、プログラムカウンタは変数  $x_3$  で、カウンタ  $a$  は変数  $x_4$  で、カウンタ  $b$  は変数  $x_5$  で表現する。

2-カウンタ機械の各基本命令を以下のようにコード化する。

コード	基本命令
0	$a := a + 1$
1	$b := b + 1$
2	$a := a - 1$
3	$b := b - 1$
4	read $a$
5	write $a$
$6 + 2L$	if $a \neq 0$ goto $L$
$7 + 2L$	if $b \neq 0$ goto $L$

2-カウンタ機械のプログラムを  $I_0, I_1, \dots, I_K$  とし、 $I_i$  のコードを  $C_i$  とする。この時、プログラムを以下のような文字列で表現する (ここで、 $c^i$  は文字  $c$  を  $i$  回繰り返した文字列を表す)。

$$0^{C_0}10^{C_1}10^{C_2}1 \dots 10^{C_K}$$

これを、プログラムのコードという。0 と 1 から成る任意の文字列は、(空のプログラムも含めて) 一つのプログラムのコードになっていることに注意。すなわち、 $\{0,1\}^*$  とプログラムの間には一対一の対応がある。

プログラム・コードを読み込むマクロは以下のように定義できる。ただし、プログラム・コードの後ろにスペースが一つ続くものとする。

- Read\_Program  $\equiv$   
clear  $x_1$ ;  $x_2 := 0$   
 $L_1$ :  $x_6 := 0$   
read  $x_0$ ; if  $x_0 = 2$  goto  $L_3$   
if  $x_0 = 1$  goto  $L_2$   
 $x_6 := x_6 + 1$ ; goto  $L_1$   
 $L_2$ :  $x_1[x_2] := x_6$ ;  $x_2 := x_2 + 1$   
goto  $L_1$   
 $L_3$ :

読み込んだプログラムを実行するマクロは以下のように定義できる。

- Execute\_Program  $\equiv$   
 $x_3 := 0$ ;  $x_4 := 0$ ;  $x_5 := 0$   
 $L_1$ : if  $x_3 \geq x_2$  goto  $L_9$   
 $x_6 := x_1[x_3]$ ;  $x_3 := x_3 + 1$   
if  $x_6 \neq 0$  goto  $L_2$   
 $x_4 := x_4 + 1$ ; goto  $L_1$   
 $L_2$ :  $x_6 := x_6 - 1$ ; if  $x_6 \neq 0$  goto  $L_3$   
 $x_5 := x_5 + 1$ ; goto  $L_1$   
 $L_3$ :  $x_6 := x_6 - 1$ ; if  $x_6 \neq 0$  goto  $L_4$   
 $x_4 := x_4 - 1$ ; goto  $L_1$   
 $L_4$ :  $x_6 := x_6 - 1$ ; if  $x_6 \neq 0$  goto  $L_5$   
 $x_5 := x_5 - 1$ ; goto  $L_1$   
 $L_5$ :  $x_6 := x_6 - 1$ ; if  $x_6 \neq 0$  goto  $L_6$   
read  $x_0$ ;  $x_4 := x_0$ ; goto  $L_1$   
 $L_6$ :  $x_6 := x_6 - 1$ ; if  $x_6 \neq 0$  goto  $L_7$   
 $x_0 := x_4$ ; write  $x_0$ ; goto  $L_1$   
 $L_7$ :  $x_6 := x_6 - 1$ ; if  $x_6 \bmod 2 \neq 0$  goto  $L_8$   
if  $x_4 = 0$  goto  $L_1$   
 $x_3 := x_6 \text{ div } 2$ ; goto  $L_1$   
 $L_8$ : if  $x_5 = 0$  goto  $L_1$   
 $x_3 := x_6 \text{ div } 2$ ; goto  $L_1$   
 $L_9$ :

インタプリタ全体は、以下のようなになる。

- Interpreter  $\equiv$   
Read\_Program  
Execute\_Program

前節で述べたように、カウンタ機械は2-カウンタ機械で模倣可能であるから、2-カウンタ機械のインタプリタを2-カウンタ機械で構成できる。

## 4 停止問題の決定不可能性

停止問題とは、カウンタ機械のプログラム  $P$  と、 $P$  への入力  $x$  が与えられたとき、 $P$  が入力  $x$  のもとで停止するかどうかを判定する問題である。

停止問題を判定するプログラムは存在するだろうか？ すなわち、 $P$  のコードと  $x$  を入力とし、 $P$  が入力  $x$  のもとで停止すれば 1 を出力し、停止しなければ 0 を出力するプログラムは存在するだろうか？

以下で述べるように、このようなプログラムは存在しないことがわかる。つまり、停止問題は決定不可能である。

このことを対角線論法を用いて証明する。

まず、停止問題を判定するプログラム  $\Pi$  が存在したとする。すなわち、 $\Pi$  は  $P$  のコードと  $x$  を入力とし、0 か 1 かを出力して停止する (無限ループにはならず、必ず停止する)。

カウンタ  $x_t, x_u$  を  $\Pi$  中に現れないカウンタとする。すると、以下のプログラム  $\Pi_1$  は  $\Pi$  と全く同一の動作を行う。

- $\Pi_1 \equiv$   
Read.Input  $x_t$   
 $x_u := 0$   
 $\Pi$  中の read  $x_0$  を以下の read'  $x_0$  で置き換える
- read'  $x_0 \equiv$   
 $x_0 := x_t[x_u]; x_u := x_u + 1$

では、次のプログラム  $\Pi_2$  は何をしているのだろうか？

- $\Pi_2 \equiv$   
Read.Input  $x_t$   
 $x_v = x_t; x_t := x_t \# " "; x_t := x_t \# x_v$   
 $x_u := 0$   
 $\Pi$  中の read  $x_0$  を read'  $x_0$  で置き換える

$\Pi_2$  への入力として、プログラム  $P$  のコード  $x$  が与えられたとしよう。すると、 $\Pi_2$  は「 $P$  が入力  $x$  のもとで停止する」ならば 1 を出力し、そうでなければ 0 を出力する (必ずどちらかを出力する)。すなわち、 $\Pi_2$  を入力  $x$  のもとで動かしたとき、以下が成り立つ。

- 1 を出力する  $\Rightarrow$  コードが  $x$  のプログラムは、入力  $x$  に対して停止する
- 0 を出力する  $\Rightarrow$  コードが  $x$  のプログラムは、入力  $x$  に対して停止しない

さらに、 $\Pi_2$  を以下のプログラム  $\Pi_3$  に書き換える。

- $\Pi_3 \equiv$   
 $\Pi_2$  中の write  $x_0$  を以下の write'  $x_0$  で置き換える
- write'  $x_0 \equiv$   
 $L_1: \text{ if } x_0 \neq 0 \text{ goto } L_1$   
 $\text{ goto } K + 1$

すなわち、write'  $x_0$  は、 $x_0$  が 1 なら無限ループになり停止せず、0 ならただちに停止する。したがって、 $\Pi_3$  を入力  $x$  のもとで動かしたとき、以下が成り立つ。

- 停止しない  $\Rightarrow$  コードが  $x$  のプログラムは、入力  $x$  に対して停止する
- 停止する  $\Rightarrow$  コードが  $x$  のプログラムは、入力  $x$  に対して停止しない

ここで、 $\Pi_3$  のコードを  $C$  とし、 $\Pi_3$  を入力  $C$  のもとで動かしたとする。どうなるだろうか？  
上の性質の  $x$  に  $C$  を代入すれば良いのだから、以下が成り立つ。 $\Pi_3$  を入力  $C$  のもとで動かしたとき、

停止しない  $\Rightarrow$  コードが  $C$  のプログラム、すなわち  $\Pi_3$  は入力  $C$  に対して停止する

停止する  $\Rightarrow$  コードが  $C$  のプログラムは、すなわち  $\Pi_3$  は入力  $C$  に対して停止しない

結局、

$\Pi_3$  が入力  $C$  に対して停止しない  $\Rightarrow$   $\Pi_3$  は入力  $C$  に対して停止する

$\Pi_3$  が入力  $C$  に対して停止する  $\Rightarrow$   $\Pi_3$  は入力  $C$  に対して停止しない

となり、矛盾である。

したがって、停止問題を判定するプログラムは存在しない。