# プログラム言語論

亀山幸義

筑波大学 情報科学学類

No. 4 (付録:停止性)



## 題材

プログラムの符号化:

- 言語 $X$ で書かれた、構文的に正しいプログラムを、自然数であらわす方法。
- 1 つのプログラムは、ASCII コードで表現される文字の列。
- 各文字は 0 から 255 までの数字で表される。それを $a, b, c, \ldots$ とする。
- それらを、$2^a \cdot 3^b \cdot 5^c \cdot \ldots$ という形の自然数とすれば、プログラム全体は 1 つの自然数に対応付く。
- なお、素因数分解の形にしたのは、自然数からプログラムに戻す時に、一意的になるようにするため。



## 停止性問題

以下の性質を持つプログラム $H$ は存在するか?

- $H$ は 2 引数関数である。
- $H(P, x)$ はどんな引数 $P, x$ に対しても、有限時間で止まり、yes か no を返す。
- プログラム $P$ が入力 $x$ に対して停止するとき、$H(P, x)$ は yes を返す。
- プログラム $P$ が入力 $x$ に対して停止しない (無限ループする) とき、$H(P, x)$ は no を返す。

このような H が存在するか、という問題が、停止性問題 (Halting Problem) である。

この章では、最終的に、「そのような H は、存在しない」ことが示される。



## 第 1 ステップ: K の定義

H が存在すると仮定すると、次の性質を満たす K がプログラムとして書けることになる。

- $K$ は 1 引数関数である。(ただし、止まらない事もあるので、正確には「1 引数の部分関数」である。)
- $H(P, P)$ が no を返すとき、$K(P)$ は yes を返す。
- $H(P, P)$ が yes を返すとき、$K(P)$ は無限ループする (値を返さない)。

再帰呼び出しなり、while ループなりを使えば、「無限ループする」ように作ることは容易である。

## 第2ステップ: K を使った推論その1

プログラム $K$ に、引数として $K$ 自身を渡すことを考える。
(Case 1) もし、$K(K)$ が停止して yes を返したら、

- $K$ の定義から、$H(K, K)$ は no を返す。
- よって、$H$ の定義から、プログラム $K$ が入力 $K$ に対して停止しない。
- よって、$K(K)$ は停止して yes を返し、かつ、停止しない、ということになり、矛盾である。

(Case 2) もし、$K(K)$ が無限ループなら、

- $K$ の定義から、$H(K, K)$ は yes を返す。
- よって、$H$ の定義から、プログラム $K$ が入力 $K$ に対して停止する。
- よって、$K(K)$ は無限ループかつ、停止する、ということになり、矛盾である。

よって矛盾である。



## 第3ステップ

$H$ がプログラムとして存在すると仮定すると、どうやっても矛盾である。すなわち、$H$ はプログラムとして存在しない。(プログラムとして書くことはできない。)

結論: 多くのプログラム言語に対して、その言語で書かれたプログラムの停止性は、決定可能ではない。



## 付録: Turing 機械とプログラム言語

Turing 機械と同等の計算能力を持つプログラム言語や計算モデルは、どんなものでも、停止性問題の解となるものは存在しない。

- (型のない) ラムダ計算の体系
- 帰納的関数の体系
- (理想化された) C 言語で書けるプログラム
- (理想化された) OCaml 言語で書けるプログラム
- (理想化された) Scheme/Lisp 言語で書けるプログラム
- (理想化された) Java 言語で書けるプログラム

なお、「Turing 機械と同等の計算能力を持つプログラム言語 (あるいは計算モデル)」のことを Turing complete (チューリングの意味で完全) と呼ぶことがある。



## 付録その2: 決定可能性

判定問題 (yes か no かを判定する問題) が決定可能とは、

- yes であるか no であるかを決定するプログラム (Turing 機械、そのほか) が存在する

ことである。上記の事からわかるように、「プログラム P が入力 x に対して停止するかどうか」を決定する判定問題は、決定可能ではない。