

計算理論 第5回
計算可能性

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2020年11月5日

最終更新：2020年11月4日 09:03

スケジュール 前半 (予定)

- 1 計算とは何か? (10/1)
- 2 計算モデル (10/8)
- 3 チャーチ・チューリングの定立 (10/15)
- ★ 休み (体育祭) (10/22)
- 4 コード化 (10/29)
- 5 計算可能性 (11/5)
- 6 停止性問題 (11/12)
- 7 再帰定理 (11/19)
- 8 前半のまとめ (11/26)

注意：予定の変更もありうる

計算不可能な部分関数

目次

- 1 計算不可能な部分関数
- 2 万能関数の計算
- 3 今日のまとめ

計算不可能な部分関数

計算不可能な部分関数：証明 (イメージ)

	0	1	2	3	4	5	6	...
P_0	↑	↓	↓	↓	↑	↓	↑	...
P_1	↑	↑	↑	↑	↑	↑	↑	...
P_2	↑	↑	↑	↑	↑	↑	↓	...
P_3	↓	↓	↓	↓	↓	↓	↓	...
P_4	↑	↑	↑	↑	↑	↑	↑	...
P_5	↓	↑	↓	↓	↓	↓	↓	...
	↓	↓	↓	↓	↓	↑

概要

この講義の主題

計算理論 (Theory of Computation)

- ▶ 計算可能性理論 (Computability Theory)
- ▶ 計算複雑性理論 (計算量理論) (Complexity Theory)

講義の進め方

- ▶ 前半：計算可能性理論 (担当：岡本)
- ▶ 後半：計算複雑性理論 (担当：垂井先生)

前回の話し と 今日の話

前回の話し

- ▶ GOTO プログラムを自然数にコード化できる (単射が存在する)
- ▶ GOTO プログラムを WHILE プログラムで模倣できる

今日の話は、この事実と考え方を使う

今日の話：この講義のハイライト (1)

- ▶ WHILE 計算不可能な部分関数が存在することの証明 (対角線論法)
- ▶ 万能プログラム (インタプリタ) の設計

計算不可能な部分関数

計算不可能な部分関数

今から、次の定理を証明する

定理：計算不可能な部分関数

計算不可能な部分関数 $f: \mathbb{N} \rightarrow \mathbb{N}$ が存在する

証明：WHILE プログラム P のコードを $enc(P)$ とする

- ▶ コード化は単射なので、 $P \neq P'$ ならば、 $enc(P) \neq enc(P')$ となる
- ▶ ∴ WHILE プログラムをそのコードの大小順に並べられる
- ▶ 並べたものを、 P_0, P_1, P_2, \dots とする

計算不可能な部分関数

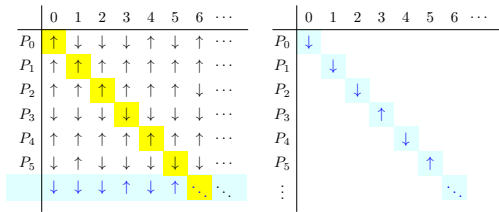
計算不可能な部分関数：証明 (続き)

- ▶ 部分関数 $diag: \mathbb{N} \rightarrow \mathbb{N}$ を次のように定義

$$diag(i) = \begin{cases} 0 & (P_i(i) \text{ が停止しないとき}) \\ \text{定義されない} & (P_i(i) \text{ が停止するとき}) \end{cases}$$

- ▶ $diag$ が WHILE 計算可能であると仮定する
- ▶ このとき、ある j に対して、 P_j が $diag$ を計算する
- ▶ $P_j(j)$ の停止性に関して、場合分けを行う
 - ▶ $P_j(j)$ が停止する $\Rightarrow diag(j) \uparrow \Rightarrow P_j(j)$ は停止しない
 - ▶ $P_j(j)$ が停止しない $\Rightarrow diag(j) = 0 \Rightarrow P_j(j)$ は停止する
- ▶ どちらの場合であっても、矛盾
- ▶ つまり、 $diag$ は WHILE 計算不可能である □

カントールの対角線論法



カントールの対角線論法

集合 A, B に対して

- ▶ A が B よりも「圧倒的に大きい」ことを証明する技法

この場合,

- ▶ $A =$ 部分関数 $\mathbb{N} \rightarrow \mathbb{N}$ 全体の集合
- ▶ $B =$ WHILE プログラム全体の集合

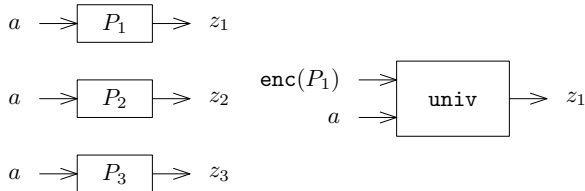
目次

① 計算不可能な部分関数

② 万能関数の計算

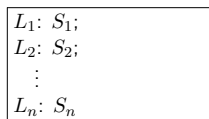
③ 今日のまとめ

万能関数：何が万能？



部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

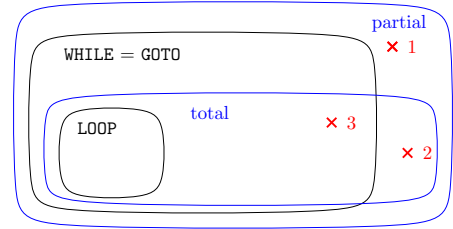
(復習) GOTO プログラムのコード化 (1) 基本アイデア



(復習) GOTO プログラムをどのようにコード化するか？

- 各文 S_1, S_2, \dots, S_n をコード化する
 $\rightsquigarrow \text{enc}(S_1), \text{enc}(S_2), \dots, \text{enc}(S_n)$
- それらと k, m を並べてリストを作る
 ▶ $k =$ 入力変数の数
 ▶ $m =$ 使用する変数の添え字の最大値
 $\rightsquigarrow (k, m, \text{enc}(S_1), \text{enc}(S_2), \dots, \text{enc}(S_n))$ (←自然数のリスト)
- そのリストをコード化する
 $\rightsquigarrow \text{enc}((k, m, \text{enc}(S_1), \text{enc}(S_2), \dots, \text{enc}(S_n)))$

(復習) 第 3 回講義のまとめ：計算可能性の比較



疑問

- 1 WHILE 計算可能でない部分関数はあるか？ 具体例はあるか？
- 2 WHILE 計算可能でない全域関数はあるか？ 具体例はあるか？
- 3 LOOP 計算可能でない全域関数はあるか？ 具体例はあるか？

→ いま解決したのは「1」

万能関数

次の部分関数 $\text{univ}: \mathbb{N}^2 \rightarrow \mathbb{N}$ を考える

$$\text{univ}(x_1, x_2) = \begin{cases} P(a_1, a_2, \dots, a_k) & (\text{下の } \boxed{\text{条件}^*} \text{ を満たすとき}) \\ \text{定義されない} & (\text{そうではないとき}) \end{cases}$$

条件*

- ▶ ある自然数 k に対して
 x_1 が k 入力 GOTO プログラム P のコードであり,
 x_2 が長さ k のリスト $a = (a_1, \dots, a_k)$ のコードであり,
 P に a を入力したときに, P が停止する

部分関数 univ を **万能関数** と呼ぶ

万能関数の計算可能性

定理：万能関数の計算可能性

万能関数 $\text{univ}: \mathbb{N}^2 \rightarrow \mathbb{N}$ は WHILE 計算可能

つまり, GOTO プログラムのインタプリタを GOTO プログラムで書ける

まず, GOTO プログラムのコード化を復習

(復習) GOTO プログラムのコード化 (2) 文のコード化

文	コード
$x_i := x_i + 1$	$\text{enc}((1, i))$
$x_i := x_i - 1$	$\text{enc}((2, i))$
GOTO L_j	$\text{enc}((3, j))$
IF $x_i = 0$ THEN GOTO L_j	$\text{enc}((4, i, j))$
HALT	$\text{enc}((5))$

長さによる帰納法で、リストのコード化を行う

- ▶ 長さ $n = 0$ のとき (リストは $()$)

$$\text{enc}(() = 0$$

- ▶ 長さ $n \geq 1$ のとき (リストは (a_1, a_2, \dots, a_n))

$$\text{enc}((a_1, a_2, \dots, a_n)) = \pi(a_1 + 1, \text{enc}((a_2, \dots, a_n)))$$

例

$$\begin{aligned} \text{enc}((3, 0, 2)) &= \pi(3 + 1, \text{enc}((0, 2))) = \pi(4, \pi(0 + 1, \text{enc}((2)))) \\ &= \pi(4, \pi(1, \pi(2 + 1, \text{enc}(())))) \\ &= \pi(4, \pi(1, \pi(3, 0))) \\ &= \pi(4, \pi(1, 6)) = \pi(4, 34) = 775 \end{aligned}$$

入力を x_1, x_2 とする

- ▶ x_2 は 長さ k のリストのコードでなくてはならない
 - ▶ $\text{isList}(x_2)$ でリストのコードか判定する
 - ▶ $k = \text{len}(x_2)$ が確認する

リストのコードが判定する関数 $\text{isList}: \mathbb{N} \rightarrow \mathbb{N}$

$$\text{isList}(x_1) = \begin{cases} 1 & (x_1 \text{ があるリストのコードであるとき}) \\ 0 & (\text{そうではないとき}) \end{cases}$$

リストの長さを返す部分関数 $\text{len}: \mathbb{N} \rightarrow \mathbb{N}$

$$\text{len}(x_1) = \begin{cases} n & (x_1 \text{ が長さ } n \text{ のリストのコードであるとき}) \\ \text{定義されない} & (\text{そうではないとき}) \end{cases}$$

変数を管理するリストを用意する

- ▶ y を長さ $m + 1$ のリストのコード $\text{enc}((0, 0, \dots, 0))$ とする
 ↳ 作り方は次のページで
- ▶ y が表すリストの 2 番目から $k + 1$ 番目の要素を x_2 が表すリストの 1 番目から k 番目の要素に書き換える
 ↳ $\text{replace}(y, j + 1, \text{elem}(x_2, j))$ を $j = 1, \dots, k$ に対して実行する

リストの i 番目の要素を変更する部分関数 $\text{replace}: \mathbb{N}^2 \rightarrow \mathbb{N}$

$$\text{replace}(e, i, x) = \begin{cases} \text{enc}((a_1, \dots, a_{i-1}, x, a_{i+1}, \dots, a_n)) & (e \text{ がリスト } (a_1, \dots, a_n) \text{ のコードであり, } \\ & 1 \leq i \leq n \text{ であるとき}) \\ \text{定義されない} & (\text{そうではないとき}) \end{cases}$$

文 S_c を実行するとき、次を行う

- ▶ e から S_c のコード s を取り出す ↳ $s := \text{elem}(e, c)$
- ▶ s がどの種類の文のコードか取り出す ↳ $v := \text{fst}(s) - 1$
 - ▶ $v = 1$ のとき, 加算
 - ▶ $v = 2$ のとき, 減算
 - ▶ $v = 3$ のとき, GOTO
 - ▶ $v = 4$ のとき, 条件つき GOTO
 - ▶ $v = 5$ のとき, HALT

文	コード
$x_i := x_i + 1$	$\text{enc}((1, i))$
$x_i := x_i - 1$	$\text{enc}((2, i))$
GOTO L_j	$\text{enc}((3, j))$
IF $x_i = 0$ THEN GOTO L_j	$\text{enc}((4, i, j))$
HALT	$\text{enc}((5))$

入力を x_1, x_2 とする

- ▶ x_1 は k 入力 GOTO プログラムのコードでなくてはならない
 - ▶ $\text{isProg}(x_1)$ で GOTO プログラムのコードか判定する
 - ▶ $k := \text{fst}(x_1) - 1$ とする
 - ▶ $m := \text{fst}(\text{snd}(x_1)) - 1$ とする

GOTO プログラム P のコード化

$$\text{enc}((k, m, \text{enc}(S_1), \text{enc}(S_2), \dots, \text{enc}(S_n)))$$

GOTO プログラムのコードが判定する関数 $\text{isProg}: \mathbb{N} \rightarrow \mathbb{N}$

$$\text{isProg}(x_1) = \begin{cases} 1 & (x_1 \text{ がある GOTO プログラムのコードであるとき}) \\ 0 & (\text{そうではないとき}) \end{cases}$$

プログラムを実行するための準備する

- ▶ $e := \text{snd}(\text{snd}(x_1))$ とする
- ▶ $c := 1$ とする (プログラムカウンタ)
 - ▶ プログラムカウンタ c が, 実行すべきラベル L_c を指定する
 - つまり, c の値に応じて, S_c を実行する
 - $c = 0$ のとき, 停止する

長さ $m + 1$ のリストのコード $\text{enc}((0, 0, \dots, 0))$ の作り方

```

m := m + 1;
y := 0; // (y = enc(()))
WHILE m DO
  y := pi(1, y)
END
    
```

$v = 1$ のとき

- ▶ $i := \text{fst}(\text{snd}(s)) - 1$ とする
- ▶ y_{i+1} を 1 だけ増やす
 ↳ $x = \text{elem}(y, i + 1); \text{replace}(y, i + 1, x + 1)$
- ▶ $c := c + 1$ とする

$v = 2$ のとき

- ▶ $i := \text{fst}(\text{snd}(s)) - 1$ とする
- ▶ y_{i+1} を 1 だけ減らす (0 のときは 0 のまま)
 ↳ $x = \text{elem}(y, i + 1); \text{replace}(y, i + 1, x - 1)$
- ▶ $c := c + 1$ とする

文	コード
$x_i := x_i + 1$	$\text{enc}((1, i))$
$x_i := x_i - 1$	$\text{enc}((2, i))$

$v = 3$ のとき

- ▶ $j := \text{fst}(\text{snd}(s)) - 1$ とする
- ▶ $c := j$ とする

$v = 4$ のとき

- ▶ $i := \text{fst}(\text{snd}(s)) - 1; j := \text{fst}(\text{snd}(\text{snd}(s))) - 1$ とする
- ▶ $y_{i+1} = 0$ のとき, $c := j$ とする
 $\rightsquigarrow \text{IF elem}(y, i + 1) = 0 \text{ THEN } c := j \text{ END}$

$v = 5$ のとき

- ▶ $c := 0$ とする

文	コード
GOTO L_j	enc((3, j))
IF $x_i = 0$ THEN GOTO L_j	enc((4, i, j))
HALT	enc((5))

- ① 計算不可能な部分関数
- ② 万能関数の計算
- ③ 今日のまとめ

今日の話：この講義のハイライト (1)

- ▶ WHILE 計算不可能な部分関数が存在することの証明 (対角線論法)
- ▶ 万能プログラム (インタプリタ) の設計

次回の予告：この講義のハイライト (2)

- ▶ WHILE 計算不可能な「意義深い」関数の具体例 (停止性)