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

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)

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

前回の話

- 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

目次

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 $f : \mathbb{N} \nrightarrow \mathbb{N}$ が存在する

証明：WHILE プログラム P のコードを $\mathrm{enc}(P)$ とする

- コード化は単射なので，$P \neq P'$ ならば，$\mathrm{enc}(P) \neq \mathrm{enc}(P')$ となる
- $\therefore$ 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，$P_0, P_1, P_2, \ldots$ とする

計算不可能な部分関数：証明 (イメージ)

	0	1	2	3	4	5	6	$\cdots$
P_0								
P_1								
P_2								
P_3								
P_4								
P_5								

計算不可能な部分関数：証明 (イメージ)

	0	1	2	3	4	5	6	$\cdots$
P_0	$\uparrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\uparrow$	$\downarrow$	$\uparrow$	$\cdots$
P_1	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\cdots$
P_2	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\downarrow$	$\cdots$
P_3	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\cdots$
P_4	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\cdots$
P_5	$\downarrow$	$\uparrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\ddots$

計算不可能な部分関数：証明 (イメージ)

	0	1	2	3	4	5	6	$\cdots$
P_0	↑	↓	↓	↓	↑	↓	↑	$\cdots$
P_1	↑	↑	↑	↑	↑	↑	↑	$\cdots$
P_2	↑	↑	↑	↑	↑	↑	↓	$\cdots$
P_3	↓	↓	↓	↓	↓	↓	↓	$\cdots$
P_4	↑	↑	↑	↑	↑	↑	↑	$\cdots$
P_5	↓	↑	↓	↓	↓	↓	↓	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\ddots$

計算不可能な部分関数：証明 (イメージ)

	0	1	2	3	4	5	6	$\cdots$
P_0	↑	↓	↓	↓	↑	↓	↑	$\cdots$
P_1	↑	↑	↑	↑	↑	↑	↑	$\cdots$
P_2	↑	↑	↑	↑	↑	↑	↓	$\cdots$
P_3	↓	↓	↓	↓	↓	↓	↓	$\cdots$
P_4	↑	↑	↑	↑	↑	↑	↑	$\cdots$
P_5	↓	↑	↓	↓	↓	↓	↓	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\ddots$

	0	1	2	3	4	5	6	$\cdots$
P_0	↓							
P_1		↓						
P_2			↓					
P_3				↑				
P_4					↓			
P_5						↑		
$\vdots$							$\ddots$	

計算不可能な部分関数：証明 (イメージ)

	0	1	2	3	4	5	6	$\cdots$
P_0	↑	↓	↓	↓	↑	↓	↑	$\cdots$
P_1	↑	↑	↑	↑	↑	↑	↑	$\cdots$
P_2	↑	↑	↑	↑	↑	↑	↓	$\cdots$
P_3	↓	↓	↓	↓	↓	↓	↓	$\cdots$
P_4	↑	↑	↑	↑	↑	↑	↑	$\cdots$
P_5	↓	↑	↓	↓	↓	↓	↓	$\cdots$
	↓	↓	↓	↑	↓	↑	$\ddots$	$\ddots$

	0	1	2	3	4	5	6	$\cdots$
P_0	↓							
P_1		↓						
P_2			↓					
P_3				↑				
P_4					↓			
P_5						↑		
$\vdots$							$\ddots$	

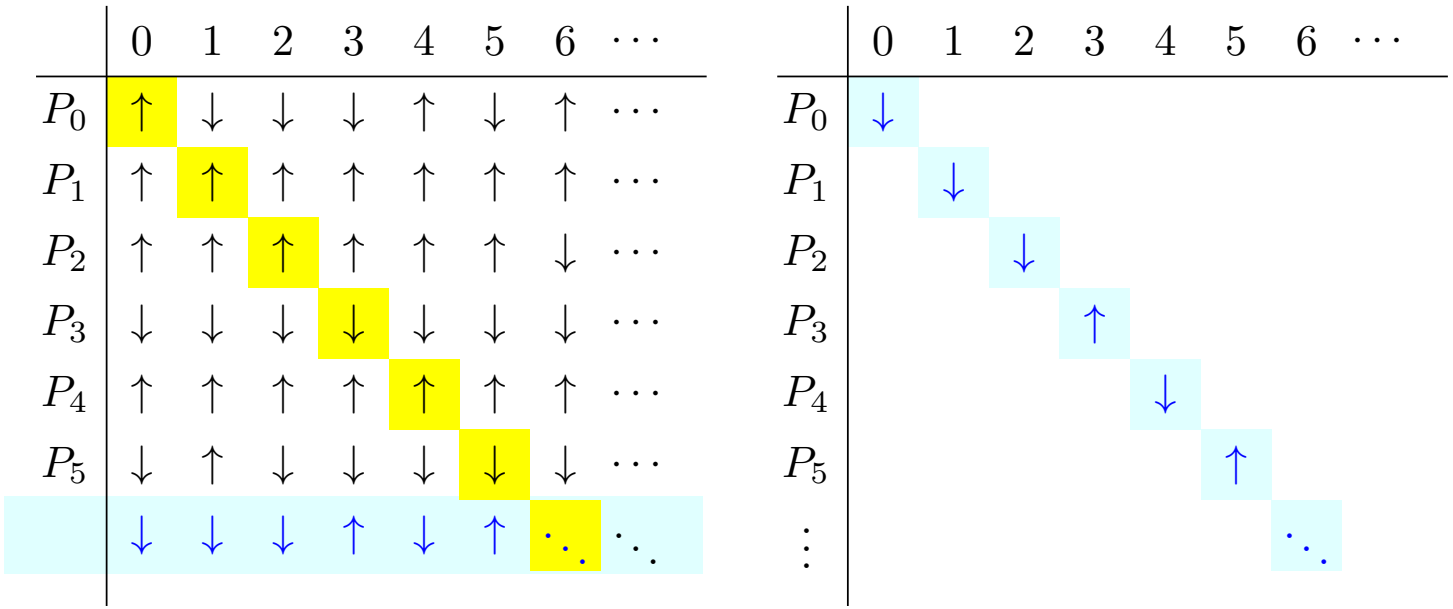
計算不可能な部分関数：証明 (続き)

- 部分関数 $\mathtt{diag}\colon \mathbb{N} \nrightarrow \mathbb{N}$ を次のように定義

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

- 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 $\Rightarrow \mathtt{diag}(j)\uparrow$ $\Rightarrow P_j(j)$ は停止しない
  - $P_j(j)$ が停止しない $\Rightarrow \mathtt{diag}(j) = 0$ $\Rightarrow P_j(j)$ 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

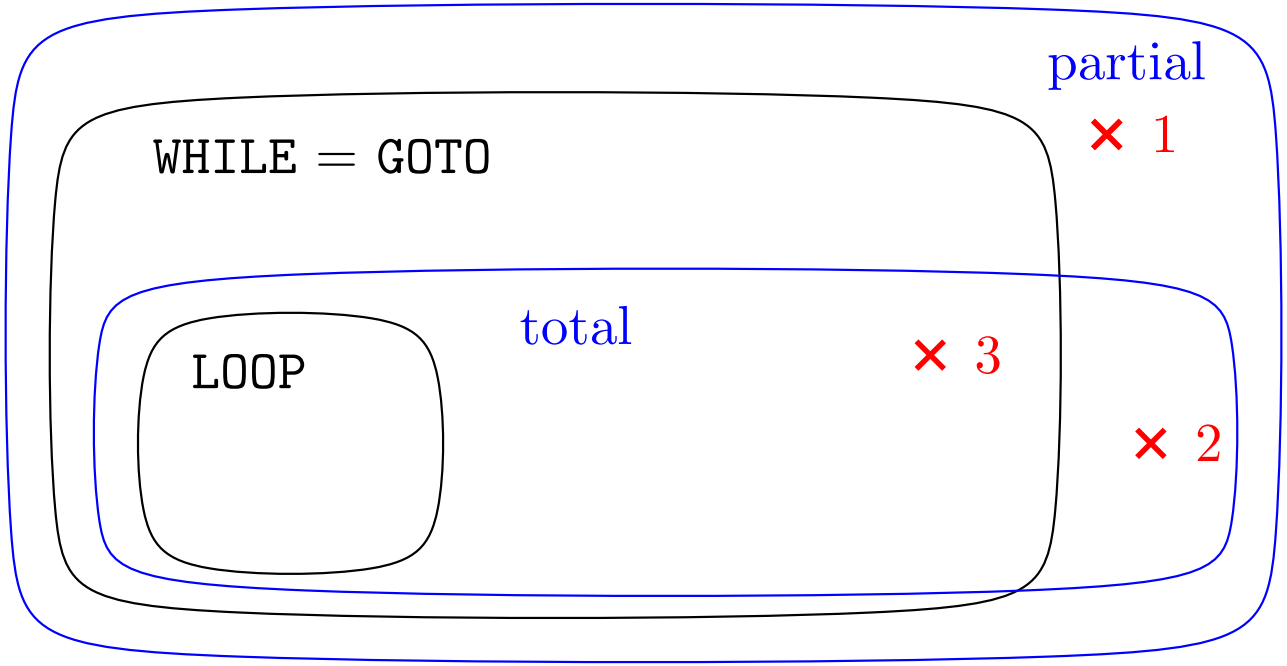
集合 A, B に対して

- 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 $\mathbb{N} \rightharpoonup \mathbb{N}$ 全体の集合
- $B =$ WHILE プログラム全体の集合

(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 」

目次

万能関数

次の部分関数 $\mathtt{univ}\colon \mathbb{N}^2 \nrightarrow \mathbb{N}$ を考える

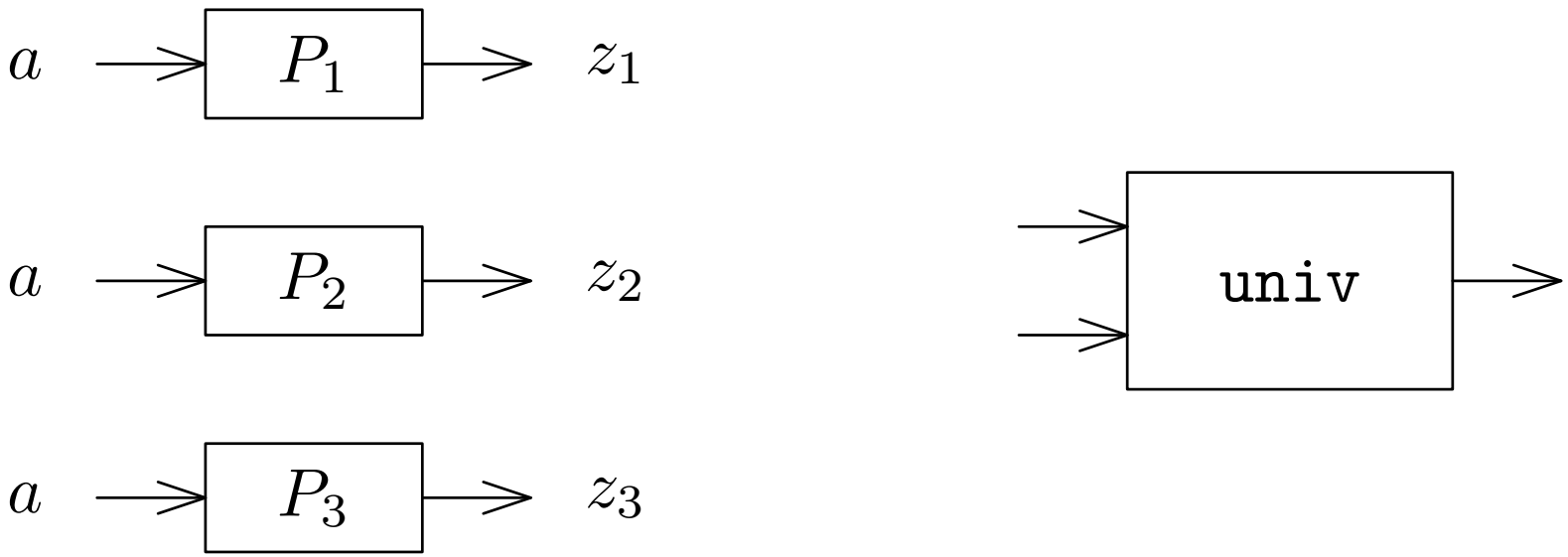
$$\mathtt{univ}(x_1, x_2) = \begin{cases} P(a_1, a_2, \ldots, a_k) & (\text{下の } \boxed{\text{条件 } *} \text{ を満たすとき}) \\ \text{定義されない} & (\text{そうではないとき}) \end{cases}$$

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 $a = (a_1, \ldots, a_k)$ のコードであり，
  P に a を入力したときに，P が停止する

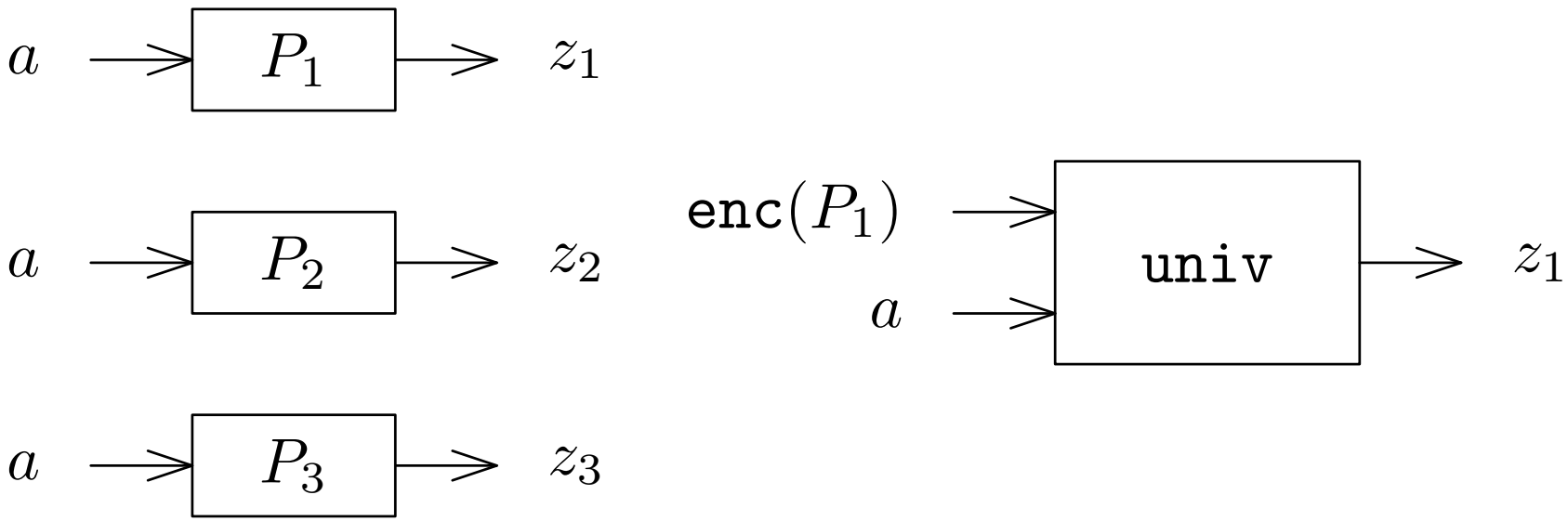
部分関数 univ を万能関数と呼ぶ

万能関数：何が万能？



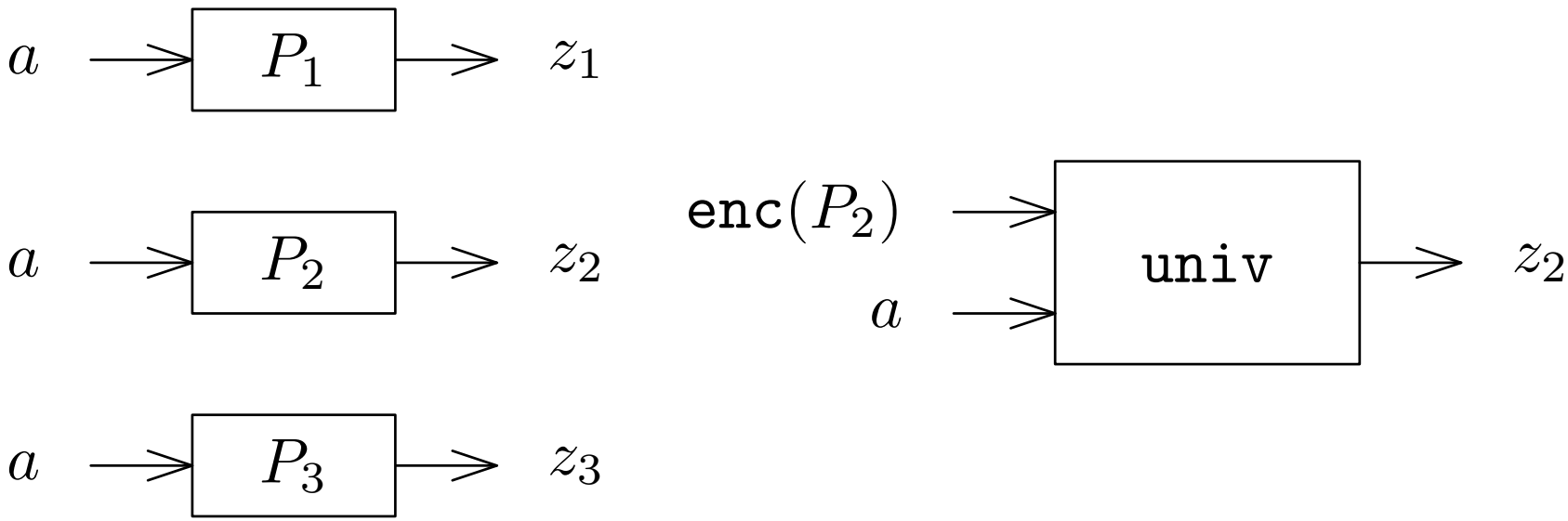
部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

万能関数：何が万能？



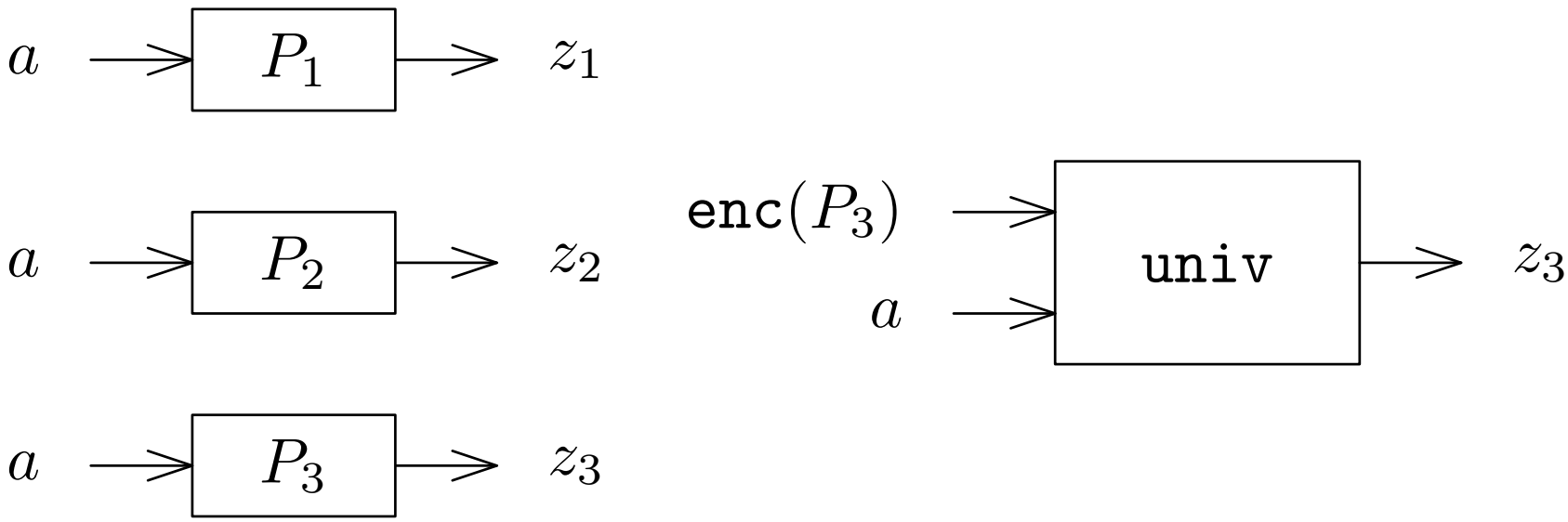
部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

万能関数：何が万能？



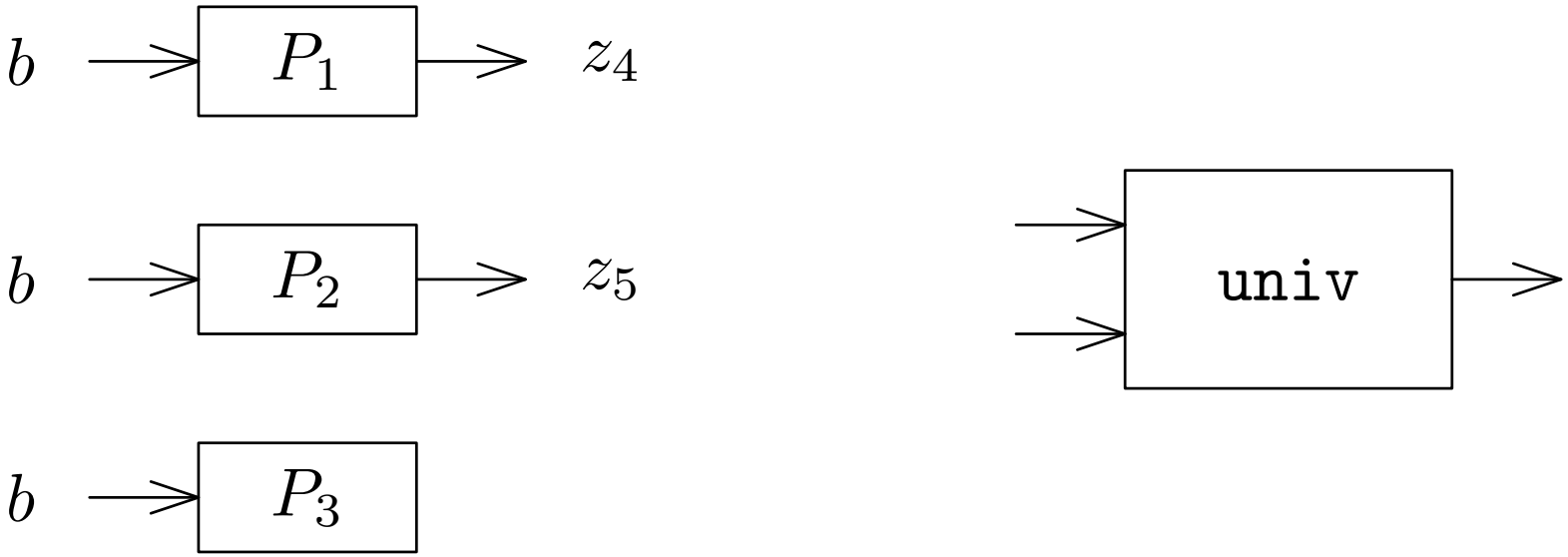
部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

万能関数：何が万能？



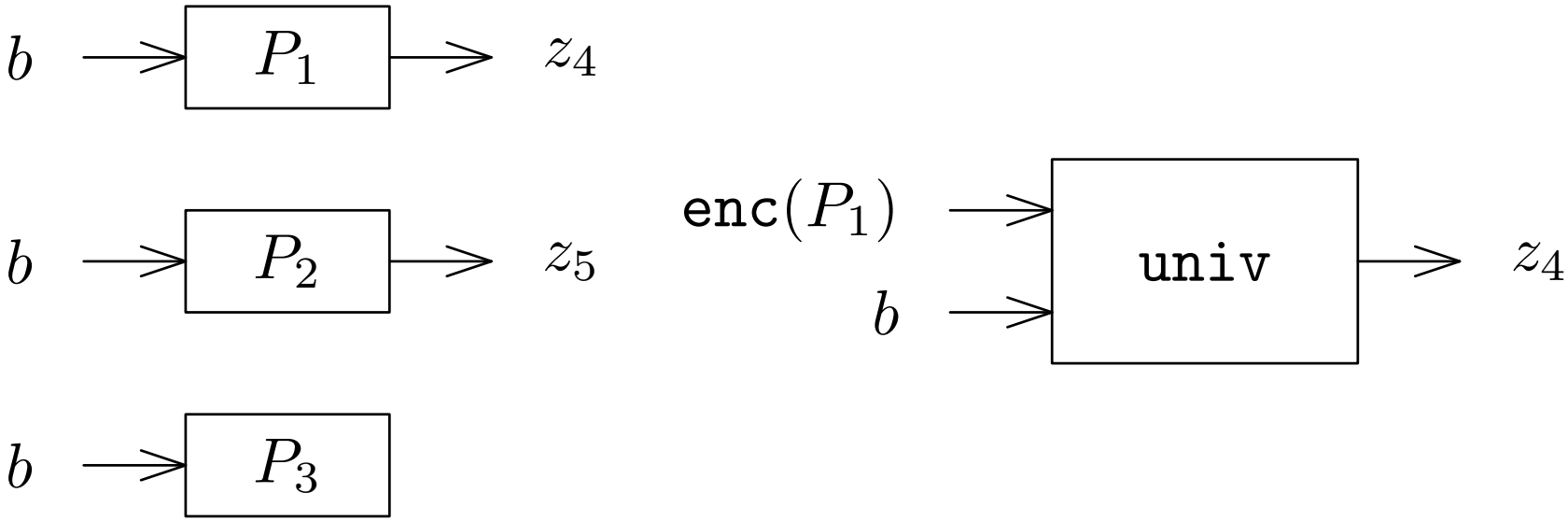
部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

万能関数：何が万能？



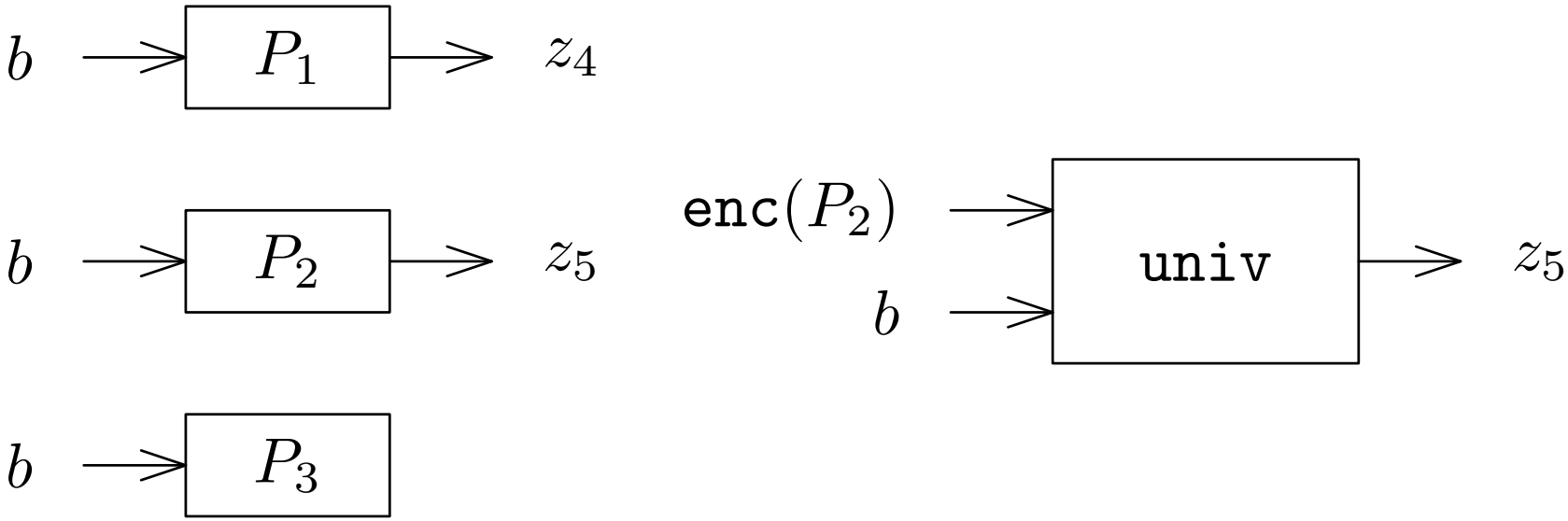
部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

万能関数：何が万能？



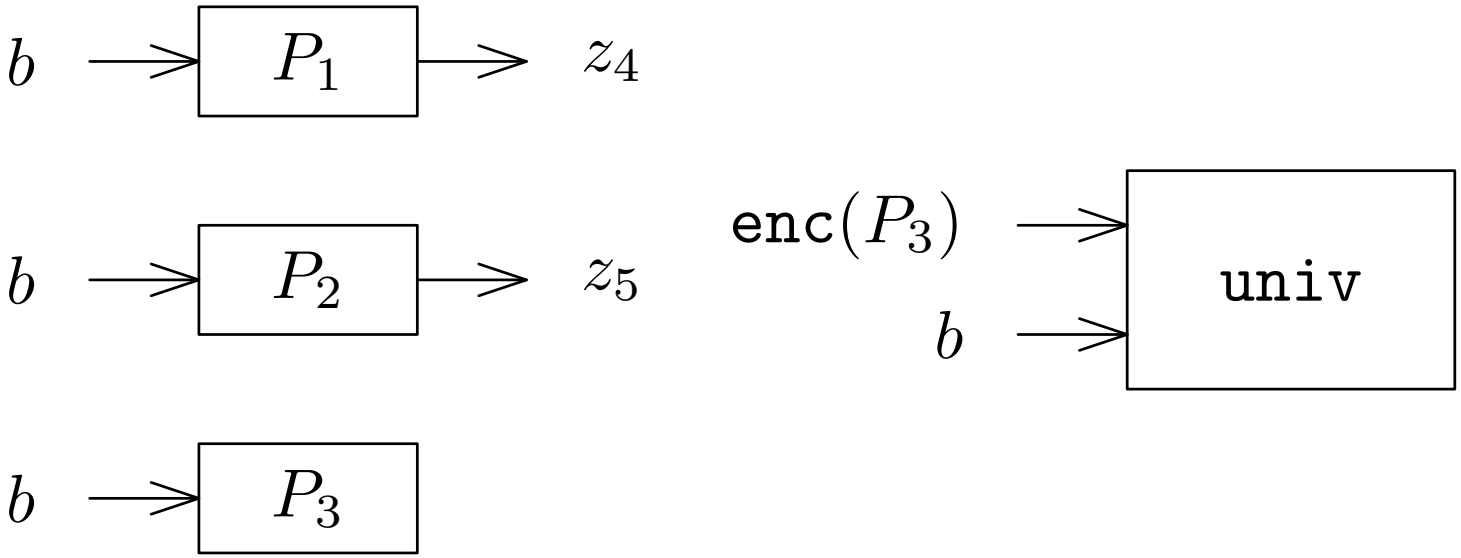
部分関数 univ は GOTO プログラムのインタプリタの役割を果たす

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

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

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 $\texttt{univ}: \mathbb{N}^2 \rightharpoonup \mathbb{N}$ 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 (1) 基本アイディア

$$
\begin{array}{l}
L_1\colon\ S_1; \\
L_2\colon\ S_2; \\
\quad\vdots \\
L_n\colon\ S_n
\end{array}
$$

(復習) GOTO プログラムをどのようにコード化するか？

1. 各文 $S_1, S_2, \ldots, S_n$ をコード化する
   $\rightsquigarrow \mathsf{enc}(S_1), \mathsf{enc}(S_2), \ldots, \mathsf{enc}(S_n)$
2. 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

   $\rightsquigarrow (k, m, \mathsf{enc}(S_1), \mathsf{enc}(S_2), \ldots, \mathsf{enc}(S_n))$ (←自然数のリスト)
3. そのリストをコード化する
   $\rightsquigarrow \mathsf{enc}((k, m, \mathsf{enc}(S_1), \mathsf{enc}(S_2), \ldots, \mathsf{enc}(S_n)))$

(復習) GOTO プログラムのコード化 (2) 文のコード化

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

(復習) リストのコード化

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 $(\ )$)

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

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

$$\mathsf{enc}((a_1, a_2, \ldots, a_n)) = \pi(a_1 + 1, \mathsf{enc}((a_2, \ldots, a_n)))$$

例

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

万能関数の計算可能性：考え方 (1)

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
  - $\mathtt{isProg}(x_1)$ で GOTO プログラムのコードか判定する
  - $k := \mathtt{fst}(x_1) - 1$ とする
  - $m := \mathtt{fst}(\mathtt{snd}(x_1)) - 1$ とする

GOTO プログラム P のコード化

$\mathsf{enc}((k, m, \mathsf{enc}(S_1), \mathsf{enc}(S_2), \ldots, \mathsf{enc}(S_n)))$

GOTO プログラムのコードか判定する関数 $\mathtt{isProg}\colon \mathbb{N} \to \mathbb{N}$

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

万能関数の計算可能性：考え方 (2)

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
  - $\texttt{isList}(x_2)$ でリストのコードか判定する
  - $k = \texttt{len}(x_2)$ か確認する

リストのコードか判定する関数 $\texttt{isList}\colon \mathbb{N} \to \mathbb{N}$

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

リストの長さを返す部分関数 $\texttt{len}\colon \mathbb{N} \rightharpoonup \mathbb{N}$

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

万能関数の計算可能性：考え方 (3)

プログラムを実行するための準備する

- $e := \mathtt{snd}(\mathtt{snd}(x_1))$ 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

万能関数の計算可能性：考え方 (4)

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 $\mathsf{enc}((0,0,\ldots,0))$ とする
  $\rightsquigarrow$ 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
  $\rightsquigarrow$ $\mathtt{replace}(y, j+1, \mathtt{elem}(x_2, j))$ を $j = 1,\ldots,k$ に対して実行する

リストの i 番目の要素を出力する部分関数 $\mathtt{elem}\colon \mathbb{N}^2 \rightharpoonup \mathbb{N}$

$$\mathtt{elem}(e,i) = \begin{cases} a_i & (e \text{ がリスト } (a_1,\ldots,a_n) \text{ のコードで} \\ & \quad \text{あり，} 1 \le i \le n \text{ のとき}) \\ \text{定義されない} & (\text{そうではないとき}) \end{cases}$$

万能関数の計算可能性：考え方 (4)

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 $\mathsf{enc}((0,0,\ldots,0))$ とする
  $\rightsquigarrow$ 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
  $\rightsquigarrow$ $\mathtt{replace}(y, j+1, \mathtt{elem}(x_2, j))$ を $j = 1,\ldots,k$ 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 $\mathtt{replace}\colon \mathbb{N}^2 \nrightarrow \mathbb{N}$

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

万能関数の計算可能性：考え方 (5)

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

$m := m+1;$
$y := 0;$ // $(y = \mathsf{enc}(( )))$
WHILE m DO
  $y := \mathtt{pi}(1, y)$
END

万能関数の計算可能性：考え方 (6)

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　$\rightsquigarrow s := \mathtt{elem}(e, c)$
- s がどの種類の文のコードか取り出す　$\rightsquigarrow v := \mathtt{fst}(s) - 1$
  - $v = 1$ 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
$x_i := x_i + 1$	$\mathsf{enc}((1, i))$
$x_i := x_i - 1$	$\mathsf{enc}((2, i))$
GOTO L_j	$\mathsf{enc}((3, j))$
IF $x_i = 0$ THEN GOTO L_j	$\mathsf{enc}((4, i, j))$
HALT	$\mathsf{enc}((5))$

万能関数の計算可能性：考え方 (7)

$v = 1$ のとき

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

$v = 2$ のとき

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

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

万能関数の計算可能性：考え方 (8)

$v = 3$ のとき

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

$v = 4$ のとき

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

$v = 5$ のとき

- $c := 0$ とする

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

目次

今日のまとめ と 次回の予告

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)

目次