

チューリングマシン

- 有限個の内部状態を持つ
- 入力データはテープ上に一区画一文字ずつ書き込まれて与えられる
- データを読み書きするヘッドがテープ上を動く
- 遷移関数は次の形：
内部状態とヘッドが今いる場所の文字とによって、その場所の文字を書き換え、次の内部状態に移り、ヘッドを左か右かに動かす
- 受理状態または拒否状態に達したら停止するが、停止しないこともある

チューリングマシンによる言語の認識

チューリングマシン T が言語 A を認識する

\Updownarrow

$$A = \left\{ w \in \Sigma^* \mid \begin{array}{l} \text{入力 } w \text{ に対し、} \\ \text{受理状態で停止する} \\ \text{遷移が存在} \end{array} \right\}$$

\Updownarrow

$w \in A \iff$ 入力 w に対し、
受理状態で停止する遷移が存在

チューリングマシンによる言語の判定

チューリングマシン T が言語 A を判定する



T は A を認識し、
かつ、全ての入力に対し必ず停止する



$w \in A \iff$ 入力 w に対し、
受理状態で停止する遷移が存在
かつ

$w \notin A \iff$ 入力 w に対し、
拒否状態で停止する遷移が存在

Church-Turing の提唱

「全てのアルゴリズム (計算手順) は、
チューリングマシンで実装できる」

(アルゴリズムと呼べるのは
チューリングマシンで実装できるものだけ)

… 「アルゴリズム」の定式化

何故「チューリングマシン」なのか？

- およそ計算機で実行したいことは模倣可能
(無限のメモリにランダムアクセスできる
計算機モデル)
- 多少モデルを変更しても強さが同じ
(モデルの頑強性)
 - ★ テープが両方に無限に伸びているか
 - ★ ヘッドが動かないことがあっても良いか
 - ★ 複数テープチューリングマシン
 - ★ 決定性 / 非決定性 などなど

万能チューリングマシン

(universal Turing machine)

全てのチューリングマシンの動作を模倣する

… プログラム内蔵方式 (von Neumann 型) の
計算機に相当

- 入力 : $(\langle M \rangle, w)$
 - ★ $\langle M \rangle$: 機械 M の符号化 (プログラムに相当)
 - ★ w : M に与える入力データ
- 出力 : 機械 M が入力 w を受理するかどうか

定理

言語

$$A_{\text{TM}} = \left\{ (\langle M \rangle, w) \mid \begin{array}{l} \langle M \rangle : \text{TM } M \text{ の符号化} \\ M \text{ が入力 } w \text{ を受理} \end{array} \right\}$$

は認識可能だが、判定可能ではない。

証明は一種の対角線論法による
(Russell のパラドックス風)

対角線論法の例：Russell のパラドックス

$X := \{A \mid A \notin A\}$ とせよ

$X \in X$ であるか？

- $X \in X$ と仮定すると、定義より $X \notin X$
- $X \notin X$ と仮定すると、定義より $X \in X$

→ どちらにしても**矛盾!!**

A_{TM} の判定不可能性

A_{TM} を判定する TM U があったとする。

入力 $\langle M \rangle$ に対し、

- M が $\langle M \rangle$ を受理するなら拒否
- M が $\langle M \rangle$ を拒否するなら受理

となる TM D が (U を使って) 作れる。

これに、入力 $\langle D \rangle$ を喰わせよ。

対角線論法の例：冪集合の濃度

集合 X の冪集合 (power set)

$$\mathcal{P}(X) = \{S \mid S \subset X\}$$

について、

$$\#X \not\leq \#\mathcal{P}(X)$$

応用： $\#\mathbb{N} = \#\mathbb{Q} = \aleph_0$ (可算集合) だが、
 $\#\mathbb{R} = \aleph \not\leq \aleph_0$ (連続体濃度)

注： \aleph は \aleph_0 の次の大きさ、とは言えない
(連続体仮説)

定理

チューリングマシンで認識可能でない言語が存在する。

- チューリングマシン全体の集合
- 言語全体の集合

の濃度とを比較せよ

さて、本講義最後の話題は、

計算量

について

問題の難しさを如何に計るか？

Church-Turing の提唱 (再掲)

「全てのアルゴリズム (計算手順) は、
チューリングマシンで実装できる」

(アルゴリズムと呼べるのは
チューリングマシンで実装できるものだけ)

… 「アルゴリズム」の定式化

計算量 (complexity)

- **時間計算量** : 計算に掛かるステップ数
(TM での計算の遷移の回数)
- **空間計算量** : 計算に必要なメモリ量
(TM での計算で使うテープの区画数)

通常は、決まった桁数の四則演算 1 回を
1 ステップと数えることが多い

入力データ長 n に対する

増加のオーダー (Landau の O -記号) で表す