チューリングマシン

離散数学・オートマトン
2023 年後期
佐賀大学理工学部　只木進一

更に強力なオートマトンが必要？

- PDA では、$\{a^n b^n c^n | n \in N\}$ を受理できない
  - スタックの制約から
  - 二つのスタックならば可能 $\rightarrow$ 自由に読み書きできるリストと同等
- 自由に読み書きできる「メモリ」をモデル化したい
- Church-Turing のテーゼ (thesis)
  - 計算できる関数とは、その関数を計算する Turing マシンが存在する関数である。
  - 計算できる、つまりアルゴリズムがあることと、Turing マシンが同等
  - 例外は知られていない

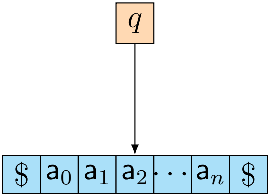
Alan Turing (1912 – 1954)

- もともとは数学者
- 第 2 次世界大戦中に、暗号解読に従事
- Manchester Mark I などの開発に従事
- Turing Test: 人工知能と人を見分ける
- 数理生物学や化学反応にも関心
  - Turing pattern など
- 「イミテーション・ゲーム」

https://www.britannica.com/biography/Alan-Turing

Turing マシン

- 読み書きできる左右に無限長のテープ
- $\$$ は、テープの空白 (何も書いていない) を表す特別な記号
  - テープへは無限に書くことができる
  - $\$$ は、その外側には何も書いていないことを表す記号
- テープヘッドは左右に動くことができる



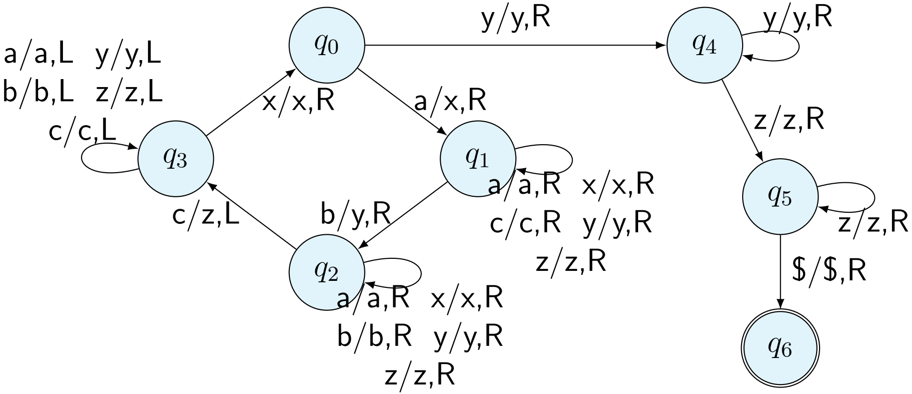
両方に無限に長いテープ

$M = \langle Q, \Gamma, \Sigma, \delta, q_0, \$, F \rangle$

- Q: 内部状態の有限集合
- Γ: テープ上のアルファベット
- $\Sigma \subseteq \Gamma$: 入力アルファベット
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\mathsf{L}, \mathsf{R}\}$
  - テープへ書き込み可能: 文字を読んだ場所に文字を上書きする
  - $\{\mathsf{L}, \mathsf{R}\}$ は、テープヘッドの左右への移動
- $q_0 \in Q$: 初期状態
- $\$ \in \Gamma \setminus \Sigma$: テープ上の空白記号: スペース記号とは異なる
- $F \subseteq Q$: 受理状態の集合

例 2.1: $L = \{a^n b^n c^n | n \in N\}$

$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\} \\
F &= \{q_6\} \\
\Gamma &= \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{x}, \mathsf{y}, \mathsf{z}, \$\} \\
\Sigma &= \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}
\end{aligned}
$$


q_0
q_1
q_2
q_3
q_4
q_5
q_6
y/y,R
a/x,R
x/x,R
b/y,R
c/z,L
a/a,L y/y,L
b/b,L z/z,L
c/c,L
a/a,R x/x,R
c/c,R y/y,R
z/z,R
a/a,R x/x,R
b/b,R y/y,R
z/z,R
y/y,R
z/z,R
z/z,R
$/$,R

動作: 状態の右側の文字を読むことに注意

$$
\begin{aligned}
q_0\text{aabbcc} &\vdash \text{x}q_1\text{abbcc} \vdash \text{xa}q_1\text{bbcc}\\
&\vdash \text{xay}q_2\text{bcc} \vdash \text{xayb}q_2\text{cc}\\
&\vdash \text{xay}q_3\text{bzc} \vdash \text{xa}q_3\text{ybzc}\\
&\vdash \text{x}q_3\text{aybzc} \vdash q_3\text{xaybzc}\\
&\vdash \text{x}q_0\text{aybzc}\\
&\vdash \cdots \vdash \text{x}q_3\text{xyyzz}\\
&\vdash \text{xx}q_0\text{yyzz} \vdash \text{xxy}q_4\text{yzz}\\
&\vdash \text{xxyy}q_4\text{zz} \vdash \text{xxyyz}q_5\text{z}\\
&\vdash \text{xxyyzz}q_5 \vdash \text{xxyyzz}q_6
\end{aligned}
$$

ここまでで、a、b、c を一つづつ x、y、z に置き換え

全ての、a、b、c を x、y、z に置き換え

動作失敗

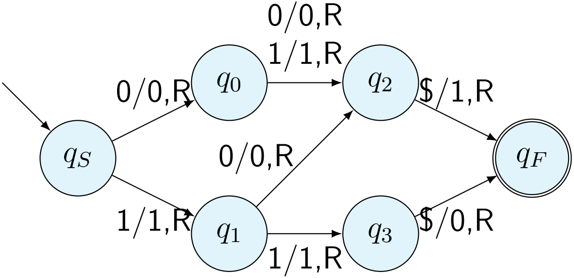
$$
\begin{aligned}
q_0\mathsf{aabbc} &\vdash \mathsf{x}q_1\mathsf{abbc} \vdash \mathsf{xa}q_1\mathsf{bbc} \\
&\vdash \mathsf{xay}q_2\mathsf{bc} \vdash \mathsf{xayb}q_2\mathsf{c} \\
&\vdash \mathsf{xay}q_3\mathsf{bz} \vdash \mathsf{xa}q_3\mathsf{ybz} \\
&\vdash \mathsf{x}q_3\mathsf{aybz} \vdash q_3\mathsf{xaybz} \\
&\vdash \mathsf{x}q_0\mathsf{aybz} \vdash \mathsf{xx}q_1\mathsf{ybz} \\
&\vdash \mathsf{xxy}q_1\mathsf{bz} \vdash \mathsf{xxyy}q_2\mathsf{z} \\
&\vdash \mathsf{xxyyz}q_2 \vdash \mathsf{xxyyz}q_2
\end{aligned}
$$

入力の受理と関数

- 例 2.1 を少し拡張
- q_6 に到達したら
  - 右端の \$ を 1 に置き換える
  - 全ての文字を \$ で置き換え、左端に 1 と書く
  - 動作失敗したら右端の \$ を 0 で置き換える
  - 全ての文字を \$ で置き換え、左端に 0 と書く
- 正しい入力は判定し、0 または 1 を返す関数に対応した Turing マシン

Turing Machine と DAND ゲート $\overline{a \wedge b}$

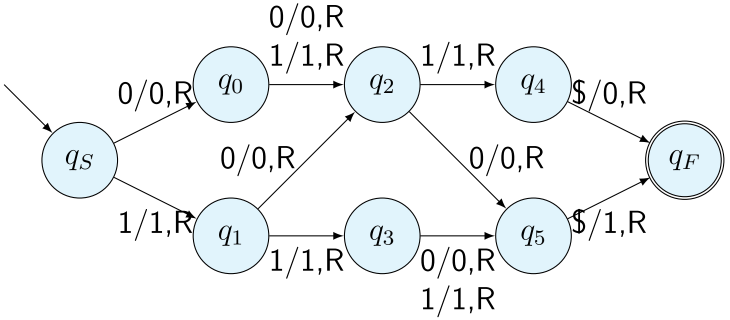
TM が「計算」できること



動作

$$
\begin{aligned}
&q_S00 \vdash 0q_00 \vdash 00q_2 \vdash 001q_F \\
&q_S01 \vdash 0q_01 \vdash 01q_2 \vdash 011q_F \\
&q_S10 \vdash 1q_10 \vdash 10q_2 \vdash 101q_F \\
&q_S11 \vdash 1q_11 \vdash 11q_3 \vdash 110q_F
\end{aligned}
$$

Turing Machine と DAND ゲート $\overline{(\overline{a \wedge b}) \wedge c}$



動作

$$
\begin{aligned}
&q_S000 \vdash 0q_000 \vdash 00q_20 \vdash 000q_5 \vdash 0001q_F\\
&q_S001 \vdash 0q_001 \vdash 00q_21 \vdash 001q_4 \vdash 0010q_F\\
&q_S010 \vdash 0q_010 \vdash 01q_21 \vdash 010q_5 \vdash 0101q_F\\
&q_S011 \vdash 0q_011 \vdash 01q_21 \vdash 011q_4 \vdash 0110q_F\\
&q_S100 \vdash 1q_100 \vdash 10q_20 \vdash 100q_5 \vdash 1001q_F\\
&q_S101 \vdash 1q_101 \vdash 10q_21 \vdash 101q_4 \vdash 1010q_F\\
&q_S110 \vdash 1q_110 \vdash 11q_30 \vdash 110q_5 \vdash 1101q_F\\
&q_S111 \vdash 1q_111 \vdash 11q_31 \vdash 111q_5 \vdash 1111q_F
\end{aligned}
$$

句構造文法: Phase Structure Grammar

- Turing マシンに対応する文法
- 生成規則

$$P : (N \cup \Sigma)^* \, N \, (N \cup \Sigma)^* \to (N \cup \Sigma)^*$$

- 文脈依存であることに注意
  - 左辺が N を必ず含む N または Σ の列

例 3.1: $L = \{a^n b^n c^n | n \in N\}$

$$N = \{S, A, B, C, D\}$$
$$\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$$

$$S \to \mathsf{a}AD, \quad A \to \mathsf{a}A\mathsf{b}B, \quad A \to C,$$
$$B\mathsf{b} \to \mathsf{b}B, \quad C\mathsf{b} \to \mathsf{b}C, \quad BD \to D\mathsf{c}, \quad CD \to \mathsf{bc}$$

導出例

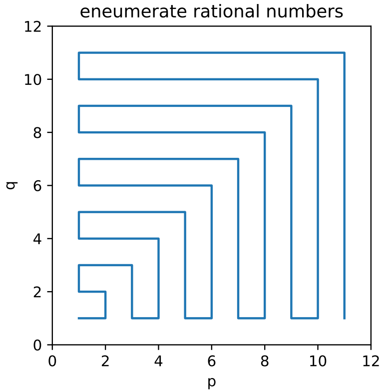
$$S \Rightarrow \mathrm{a}AD \Rightarrow \mathrm{a}CD \Rightarrow \mathrm{abc}$$

$$S \Rightarrow \mathrm{a}AD \Rightarrow \mathrm{aa}A\mathrm{b}BD \Rightarrow \mathrm{aa}C\mathrm{b}BD \Rightarrow \mathrm{aa}C\mathrm{b}D\mathrm{c}$$
$$\Rightarrow \mathrm{aab}CD\mathrm{c} \Rightarrow \mathrm{aabbcc}$$

$$S \Rightarrow \mathrm{a}AD \Rightarrow \mathrm{aa}A\mathrm{b}BD \Rightarrow \mathrm{aaa}A\mathrm{b}B\mathrm{b}BD$$
$$\Rightarrow \mathrm{aaa}C\mathrm{b}B\mathrm{b}BD \Rightarrow \mathrm{aaab}CB\mathrm{b}BD$$
$$\Rightarrow \mathrm{aaab}C\mathrm{b}BBD \Rightarrow \mathrm{aaabb}CBBD$$
$$\Rightarrow \mathrm{aaabb}CBD\mathrm{c} \Rightarrow \mathrm{aaabb}CD\mathrm{cc} \Rightarrow \mathrm{aaabbbccc}$$

無限を数える: 正の有理数を列挙する

- 有理数と自然数は同じ数だけ存在する
- 全ての有理数に異なる自然数を対応づけることができる



無限を数える: 無理数は列挙できない

- $x \in [0, 1)$ が列挙できると仮定

$$
\begin{aligned}
1 &\leftrightarrow 0.a_{11}a_{12}a_{13}a_{14}\cdots \\
2 &\leftrightarrow 0.a_{21}a_{22}a_{23}a_{24}\cdots \\
3 &\leftrightarrow 0.a_{31}a_{32}a_{33}a_{34}\cdots \\
4 &\leftrightarrow 0.a_{41}a_{42}a_{43}a_{44}\cdots
\end{aligned}
$$

- $y = 0.b_1b_2b_3b_4\cdots(b_i \neq a_{ii})$ は列挙したリストに含まれない
  - 列挙できるならば、上記リストに含まれている
- 列挙できるという仮定と矛盾
- 対角線論法 (diagonal method)

Gödel ナンバリング

- Turing マシンを列挙する
  - アルファベット、状態を整数と対応付け
  - 遷移関数は、整数から整数への写像
  - Turing マシンも整数に対応させることができる
- $M = \langle Q, \{0,1\}, \Gamma, \Sigma, \delta q_1, \$, F \rangle$
  - $Q = \{q_1, q_2, \cdots\}$
  - $\Gamma = \{x_1, x_2, \cdots\}$
  - $D = \{\mathsf{L}, \mathsf{R}\} = \{D_1, D_2\}$
- $\delta(q_i, x_j) = (q_k, x_\ell, D_m)$ に対して

$$0^i 1 0^j 1 0^k 1 0^\ell 1 0^m$$

万能 Turing マシン

- Turing マシンの動作を模倣する万能 Turing マシンが存在できる
  - Turing マシンは符号化できる
  - Turing マシンとその入力を受け取り、その動作を模倣する

Turing マシンの停止問題

- Turing マシンを整理
  - テープ上の入力に対して、結果をテープに残す。
  - テープに残ったものが関数の値。
  - テープ上の入力に対する関数と考える
- 必ず停止するか?
- これから問題とするのは
  - 答えはあるのに、計算で答えを求められない問題の存在
  - 答えを得るためのアルゴリズムがない問題の存在

決定問題: decision problem

- 答えが true/false のいずれかである関数・問題: 述語
- 例:$x^2 + y^2 = z^2$ を満たす自然数の組 (x, y, z) は存在するか?
  - (x, y, z) は列挙可能であるため、順に生成できる
  - $x^2 + y^2 = z^2$ に代入し、等号が成立する場合に、true を返して、停止
  - 例えば、$(x, y, z) = (3, 4, 5)$ を見つけて停止

決定可能:decidable

- ある Turing マシン M が、決定問題 P の具体例 w に対して、答えを出して停止する
  - 前のシートの例は、停止する
- $x^3 + y^3 = z^3$ を満たす自然数の組 $\{x, y, z\}$ を見つける問題では、前のシートの方法では停止しない
  - Fermat の最終定理: $x^n + y^n = z^n$ $(n > 2)$ を満たす整数の組 $\{x, y, z\}$ は存在しない

停止問題: Halting Proglem

- 任意の Turing マシン M に対して、入力 w を与えると停止するか
  - これ自体が述語 $f(M, w)$
- Turing マシンの停止問題は決定不能
  - true/false を決定できない
  - false と停止しないことは違うことに注意

- Turing マシンとその入力は列挙できることに注意

$$a_{ij} = \begin{cases} 1 & M_i\text{は、入力 } x_j\text{に対して停止} \\ 0 & M_i\text{は、入力 } x_j\text{に対して停止しない} \end{cases}$$

- 停止問題を解く Turing マシン $\tilde{M}$ が存在すると仮定
  - 任意の x_i に対して、$a_{ii} = 0$ のとき、かつそのときだけ停止する Turing マシン M_d に対して、$\tilde{M}$ が停止を判断できる
  - M_d 自体が、列挙した M_i のいずれか
- M_d を M_j とする
  - $a_{jj} = 1$ ならば、M_j は停止するが、M_d は停止しないことになる
  - $a_{jj} = 0$ ならば、M_j は停止ぜず、M_d は停止することになる
- 矛盾するため、$\tilde{M}$ は存在できない

停止問題が決定不能とは

- 停止問題
  - $f(w)$ の値を決定する
- 停止問題が決定不能
  - 述語 $f(w)$ の真偽を判定できない場合がある
- 正しく設定された述語 $f(w)$ は、真か偽のいずれかである。
  - しかし、判定できない場合がある
- 数学のような厳密な論理体系にあっても、計算によって証明できない命題が存在し得る