



↑
スライドはここに置いてあります

サマースクール数理物理

計算量理論入門 — 完全問題と「複雑さ」 —

河村彰星

令和7年8月29日～31日

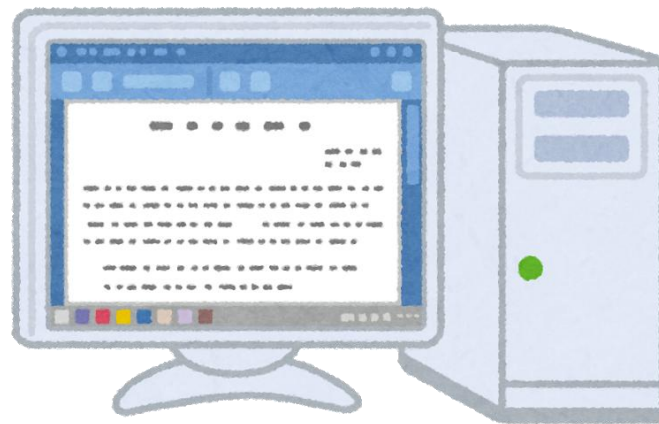
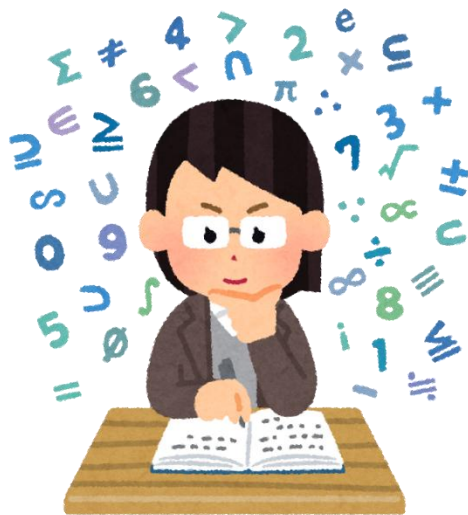




計算量理論

(Computational) Complexity Theory

「どれほど計算し難いか」という尺度で 複雑さを測る





第一日 計算可能性



計算して問題を解くことの難しさを論じたいので……

まず **問題**とは？

ここでは

各入力に対して 答が○か×か定めたもの
(入力は文字列で表される)

と考えることにする

例えば

問題

PRIME

0 1 … 9 からの文字列

与えられた正整数 (十進法で書く) が素数か答えよ

入力	1	2	3	4	5	6	7	…
	↓	↓	↓	↓	↓	↓	↓	
答	×	○	○	×	○	×	○	…

a と b からの文字列

問題

ABA

与えられた文字列に「aba」が現れるか答えよ



計算して問題を解くことの難しさを論じたいので……

まず **問題**とは？

ここでは

各入力に対して 答が○か×か定めたもの
(入力は文字列で表される)

と考えることにする

例えば

0 1 … 9 からなる文字列

問題

PRIME

与えられた正整数 (十進法で書く) が素数か答えよ

入力	1	2	3	4	5	6	7	…
	↓	↓	↓	↓	↓	↓	↓	
答	×	○	○	×	○	×	○	…



問題です。57 は素数でしょうか？

それは**入力**であって**問題**ではない！



計算理論警察

問題とは
コレ全体のこと！



計算して問題を解くことの難しさを論じたいので……

まず 問題とは？

定義

有限個の文字からなる集合 Σ を考える

Σ に属する文字を有限個並べて得られる文字列の全体を Σ^* と書く

Σ^* の部分集合を **言語** という

文字列 u の後に v を付けた文字列を uv
 文字列 $\underbrace{uu \cdots u}_{m \text{ 個}}$ を u^m
 などと表すことにする

この講義では 与えられた文字列が言語に属するか問う問題のみを考える

例 文字集合 $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ を考える (今後は一々断らず適当に定める)

Σ 上の文字列 (36 とか 119 とか ^{空文字列} ε とか) の全体を Σ^* で表す

文字列のうち十進法で素数を表すものの全体が

言語 $\text{PRIME} = \{2, 3, 5, 7, 11, 13, \dots\} \subseteq \Sigma^*$ である

この講義では「文字列を入力すると それが PRIME に属するか
 答えてくれる計算機械」などについて考えたい

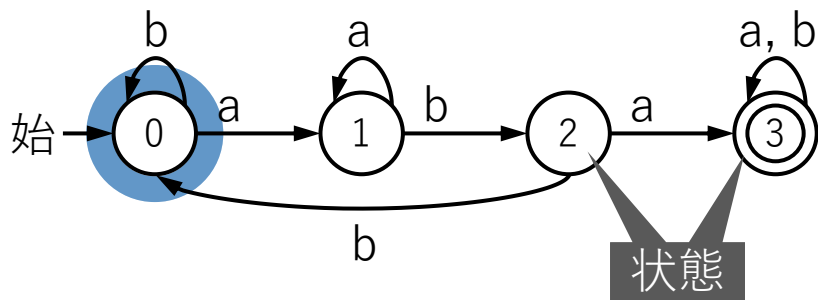
この「入力された文字列が言語 PRIME に属するか判断せよ」という問題
 のことも PRIME と呼ぶ (言語と問題は同じものとする) ことにする



問題 ABA

与えられた文字列に「aba」が現れるか答えよ

この問題を解く有限状態機械



a b b a a b a b



受理

a b b a a b b a



不受理

状態は有限個
矢印も有限本

➡ 有限の記述で書き表せる

現在の状態

	0	1	2	③
読んだ文字				
a	1	1	3	3
b	0	2	0	3

(次にどの状態に行くか記した表)



定義

有限状態機械は次のものにより指定される

- 有限個の**状態**の集合 Q 但し次のものが定まっている
 - 始状態 $q_{\text{始}} \in Q$
 - 受理状態の集合 $Q_{\text{受理}} \subseteq Q$
- 有限個の**文字**の集合 Σ
- **遷移規則**と呼ばれる関数 $\delta: Q \times \Sigma \rightarrow Q$

初め機械は始状態 $q_{\text{始}}$ にあり 与えられた文字列 $x \in \Sigma^*$ の左端から
次のこと（遷移）を繰り返す

状態 q で文字 σ を読むと

状態を $\delta(q, \sigma)$ に変えて一つ右の文字に進む

右端まで終ったとき状態が $Q_{\text{受理}}$ に属すれば機械は x を**受理**したという



定義

機械 M が言語 A を**認識**するとは 任意の入力 x に対し

- $x \in A$ のとき M は x を受理し かつ
- $x \notin A$ のとき M は x を受理しない

すべての入力で
正しく判断！

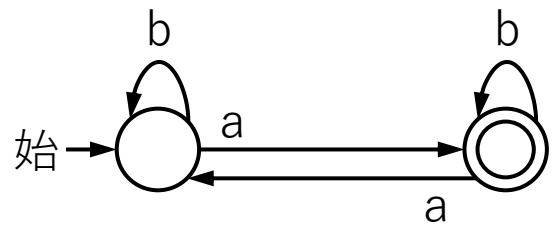
入力		ϵ	a	b	aa	ab	ba	bb	aaa	...	
問題	が要求する	答	○	×	×	○	○	×	○	×	...
機械	による計算の結果		受理	不受理	不受理	受理	受理	不受理	受理	不受理	

演習

a と b からなる文字列のうち

- (1) a が奇数回現れるものの全体
- (2) 左から 3 文字目が a であるものの全体
- (3) 右から 3 文字目が a であるものの全体

をそれぞれ認識する有限状態機械を作ってください



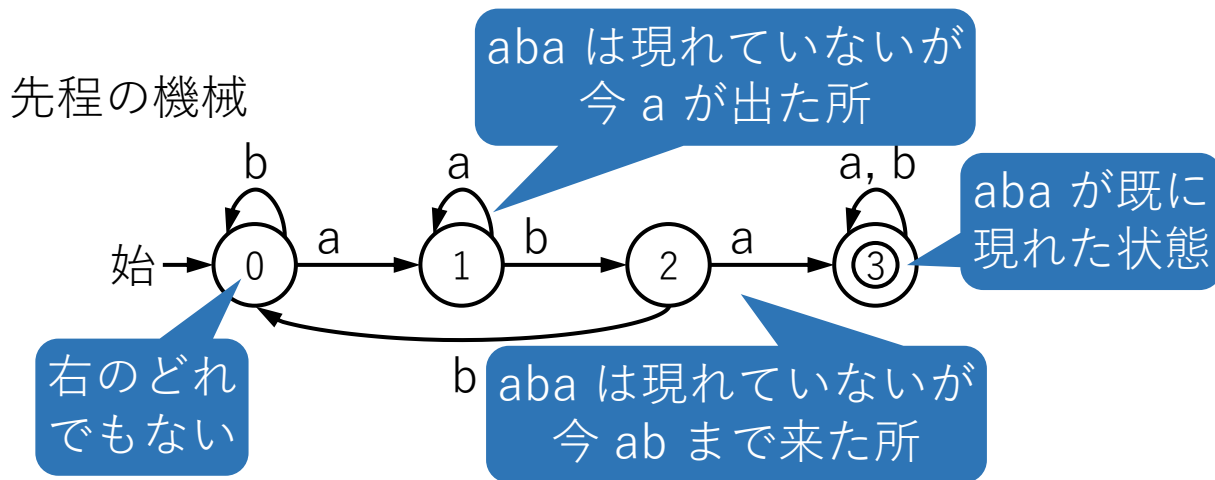
(1) の答 (例)



有限状態機械の限界

ABA はなぜ認識できたか？

各時点で「今まで読んだ部分について覚えておくべき情報」が有限種類しかないから



そういう単純な言語しか認識できない

例えば次の言語は無理



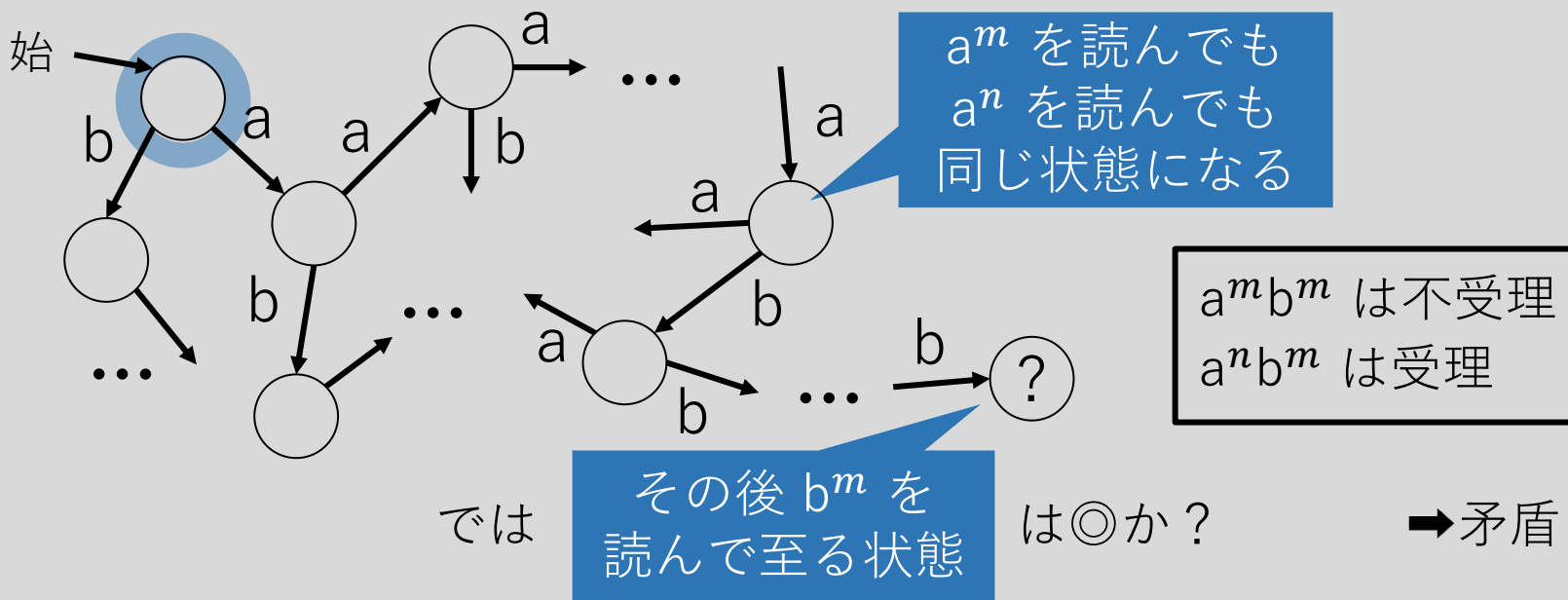
問題 MOREA

与えられた文字列に a が b よりも多く現れるか答えよ

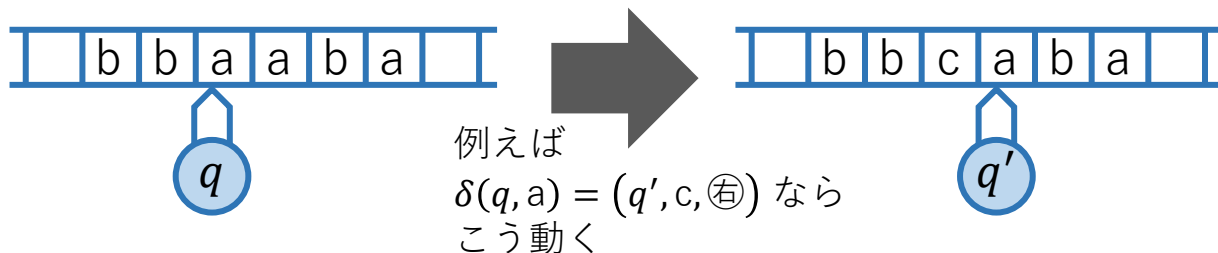
定理

MOREA を認識する有限状態機械は存在しない

証明 そのような機械があるとする
状態は有限個なので 或る数 m と n ($m < n$ とする) が存在して



読むばかりでなく
書く機能（と
左右に動く機能）
があれば？



定義

チューリング機械は次のものにより指定される

- 有限個の**状態**の集合 Q 但し次のものが定まっている
 - 始状態 $q_{\text{始}} \in Q$
 - 受理状態の集合 $Q_{\text{受理}} \subseteq Q$
- 有限個の**文字**の集合 Σ これと空白文字 $_$ を含む集合 $\Gamma \supseteq \Sigma \cup \{_$
- **遷移規則** $\delta: Q \times \Gamma \rightarrow (Q \times \Gamma \times \{\text{左}, \text{右}\}) \cup \{\text{止}\}$

初め機械は始状態 $q_{\text{始}}$ にあり テープ上に与えられた文字列 $x \in \Sigma^*$ の左端から始めて 次のこと（遷移）を繰り返す

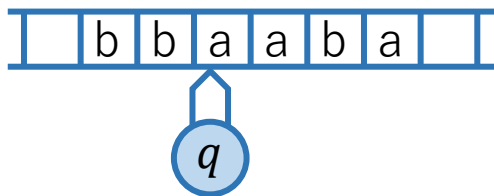
状態 $q \in Q$ で文字 σ を読むと $\delta(q, \sigma)$ が

- **止** ならば停止する
- (q', σ', d) なら 状態を q' にし σ' を書込み d の向きに一歩進む

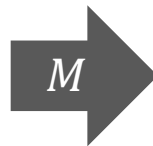
$Q_{\text{受理}}$ に属する状態で停止したら機械は x を**受理**したという

これを厳密に
定義すると……

※停止せず永遠に動き続ける場合は 受理していないと考える

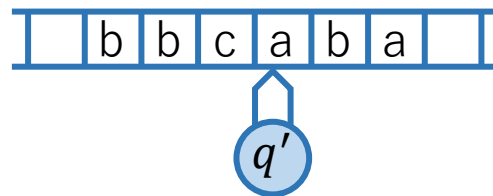


この状況を $bbqaaba$
のように表すことにする



状況の遷移

例えば $\delta(q, a) = (q', c, \text{Ⓢ})$ なら...



次の状況 $bbcq'aba$

Γ^* の文字列の途中に Q の元がちょうど 1 回だけ現れる文字列を**状況**と呼び (但し先頭や末尾に $\sqcup$ を加えた文字列も同じ状況とみなす) 状況の遷移 $\xrightarrow{M}$ を次で定義する

状態 $q \in Q$ と文字 $\sigma \in \Gamma$ に対し

- もし $\delta(q, \sigma) = (q', \sigma', \text{Ⓢ})$ ならば 任意の $u, v \in \Gamma^*$ と $\tau \in \Gamma$ に対し $u\tau q\sigma v \xrightarrow{M} uq'\tau\sigma'v$
- もし $\delta(q, \sigma) = (q', \sigma', \text{Ⓢ})$ ならば 任意の $u, v \in \Gamma^*$ に対し $uq\sigma v \xrightarrow{M} u\sigma'q'v$

文字列 $x \in \Sigma^*$ を機械 M が**受理**するとは

状況の有限列 $(s_0, \dots, s_f)$ であって次を満たすものが存在することをいう

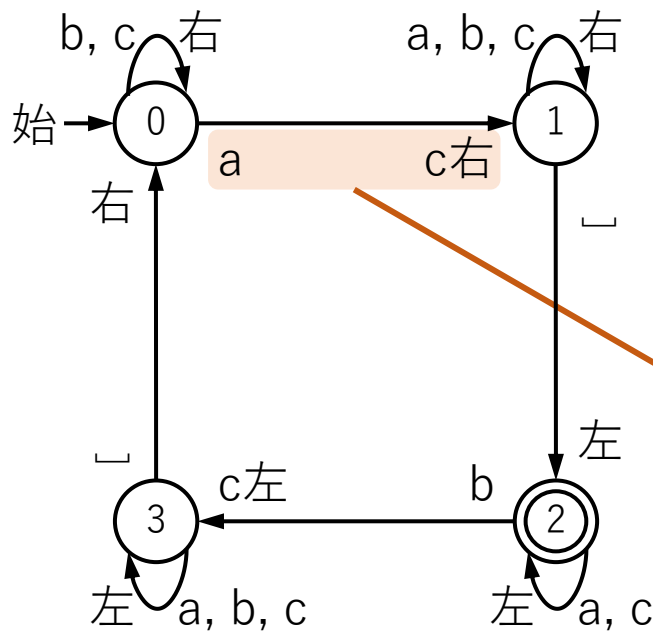
- $s_0 = q_{\text{始}}x$
- $s_0 \xrightarrow{M} s_1 \xrightarrow{M} s_2 \xrightarrow{M} \dots \xrightarrow{M} s_f$
- s_f には $\delta(q, \sigma) = \text{止}$ かつ $q \in Q_{\text{受理}}$ なる $q\sigma$ が現れる



問題
MOREA

与えられた文字列に
a が b よりも多く現れるか答えよ

MOREA を認識する
チューリング機械

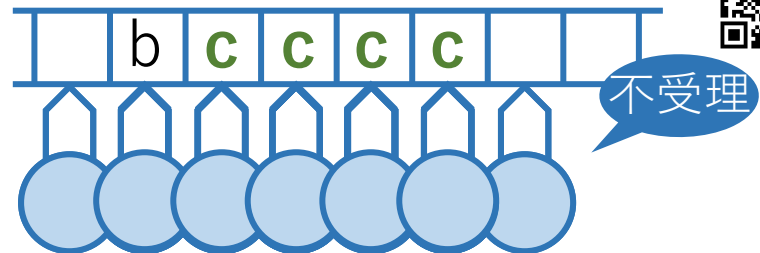


- 状態 0 で
文字 a を読むと
- 状態を 1 にし
 - 文字 c を書込み
 - 右へ行く

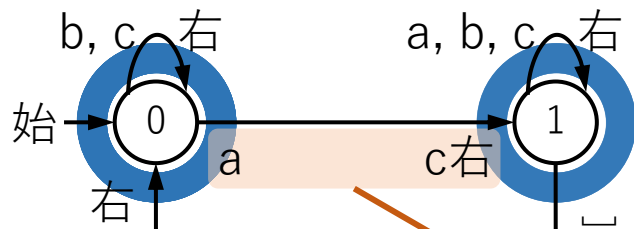


問題
MOREA

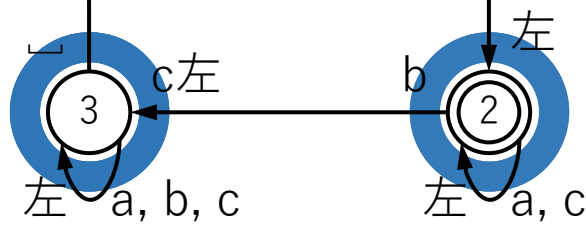
与えられた文字列に
a が b よりも多く現れるか答えよ



初めは状態 0
右に読み進め 最初の a を
c に書き換えて状態 1 になり
そのまま右端まで進む



右端に達すると状態 2 になって
左へ戻ってゆき 最初の b を
c に書き換えて状態 3 になり
そのまま左端まで進む



状態 0 で
文字 a を読むと

- 状態を 1 にし
- 文字 c を書込み
- 右へ行く

〔 辿れる矢印が
ないときは止 〕

読取りのみでは解けない問題を解けるようになった！



定義

(読取りのみの)

言語 A を認識する有限状態機械が存在するとき
 A は**正則**であるという

regular

定義

言語 A を認識するチューリング機械が存在するとき
 A は**認識可能**である ($A \in \mathbf{CE}$) という

computably enumerable
(recursively enumerable)

• ABA

正則

• MOREA

CE



チャーチとチューリングの定立

「計算できる」
(機械的な手順で解ける)

=

(チューリング機械で)
認識可能

ボンヤリ
した概念

数学的にハッキリ
定義した概念



チューリング機械ですべての「計算」が実現できているなんて本当？

幾つかの理由から 今では広く受け入れられています

- 現実の計算に出て来そうな色々な手順は
確かにチューリング機械で書けるようだ (経験上)
- 他の様々なやり方で定義された計算可能性の概念とも一致



チューリングの論文 (次頁)

A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* 2, **42**: 230–265, 1936.

1. [Type (α)]. This argument is only an elaboration of the ideas of § 1.



Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book. In elementary arithmetic the two-dimensional character of the paper is sometimes used. But such a use is always avoidable, and I think that it will be agreed that the two-dimensional character of paper is no essential of computation. I assume then that the computation is carried out on one-dimensional paper, *i.e.* on a tape divided into squares. I shall also suppose that the number of symbols which may be printed is finite. If we were to allow an infinity of symbols, then there would be symbols differing to an arbitrarily small extent[†]. The effect of this restriction of the number of symbols is not very serious. It is always possible to use sequences of symbols in the place of single symbols. Thus an Arabic numeral such as

紙に文字を書き
ながら計算する
様子を考えよう

文字は有限個と
考えてよからう

無限個使っても
区別できなきゃ
意味ないので

[†] If we regard a symbol as literally printed on a square we may suppose that the square is $0 \leq x \leq 1$, $0 \leq y \leq 1$. The symbol is defined as a set of points in this square, *viz.* the set occupied by printer's ink. If these sets are restricted to be measurable, we can define the "distance" between two symbols as the cost of transforming one symbol into the other if the cost of moving unit area of printer's ink unit distance is unity, and there is an infinite supply of ink at $x = 2$, $y = 0$. With this topology the symbols form a conditionally compact space.



17 or 9999999999999999 is normally treated as a single symbol. Similarly in any European language words are treated as single symbols (Chinese, however, attempts to have an enumerable infinity of symbols). The differences from our point of view between the single and compound symbols is that the compound symbols, if they are too lengthy, cannot be observed at one glance. This is in accordance with experience. We cannot tell at a glance whether 9999999999999999 and 9999999999999999 are the same.

大きな数値を
記号と考えても
やはり瞬時には
区別できない

The behaviour of the computer at any moment is determined by the symbols which he is observing, and his “state of mind” at that moment. We may suppose that there is a bound B to the number of symbols or squares which the computer can observe at one moment. If he wishes to observe more, he must use successive observations. We will also suppose that the number of states of mind which need be taken into account is finite. The reasons for this are of the same character as those which restrict the number of symbols. If we admitted an infinity of states of mind, some of them will be “arbitrarily close” and will be confused. Again, the restriction is not one which seriously affects computation, since the use of more complicated states of mind can be avoided by writing more symbols on the tape.

現在の文字と
現在の状態から
次の動きを決める

状態も有限個と
考えてよからう

一度に読み書き
できる文字数も



チューリング機械と（適当な符号化の下で）等価な計算可能性の定義としては例えば次の「再帰的函数」がある

定義

自然数の組から自然数への部分函数 $f: \subseteq \mathbf{N}^n \rightarrow \mathbf{N}$ ($n = 0, 1, 2, \dots$) が

再帰的 (recursive) であることを次で定める

- 次で定義される $\text{zero}: \mathbf{N}^0 \rightarrow \mathbf{N}$ と $\text{proj}_i^n: \mathbf{N}^n \rightarrow \mathbf{N}$ と $\text{succ}: \mathbf{N} \rightarrow \mathbf{N}$ は再帰的

$$\text{zero}(\) = 0 \quad \text{proj}_i^n(x_1, \dots, x_n) = x_i \quad \text{succ}(x) = x + 1$$

- $g: \subseteq \mathbf{N}^n \rightarrow \mathbf{N}$ と $h_1, \dots, h_n: \subseteq \mathbf{N}^m \rightarrow \mathbf{N}$ とが再帰的ならば

次で定義される $f: \subseteq \mathbf{N}^m \rightarrow \mathbf{N}$ も再帰的

$$f(x) = g(h_1(x), \dots, h_n(x))$$

- $g: \subseteq \mathbf{N}^n \rightarrow \mathbf{N}$ と $h: \subseteq \mathbf{N}^{n+2} \rightarrow \mathbf{N}$ が再帰的ならば

次で定義される $f: \subseteq \mathbf{N}^{n+1} \rightarrow \mathbf{N}$ も再帰的

$$f(0, y) = g(y)$$

$$f(x + 1, y) = h(x, y, f(x, y))$$

- $g: \mathbf{N}^{n+1} \rightarrow \mathbf{N}$ が再帰的ならば 次で定義される $f: \subseteq \mathbf{N}^n \rightarrow \mathbf{N}$ も再帰的

$$f(x) = \min\{y \in \mathbf{N} \mid g(x, y) = 0\}$$

但し各等式の左辺は「右辺が順次定義されるとき定義される」と解釈する
(例えば最後のものでは $g(x, 0), \dots, g(x, y - 1)$ がみな定義されて 0 でなくかつ $g(x, 0) = 0$ であるような y が存在するときに $f(x)$ が定義される)



でも文字列に関する
問題しかないの？

入力を**符号化**する方法を決めた上で
文字列に関する問題として扱います

問題
PRIME

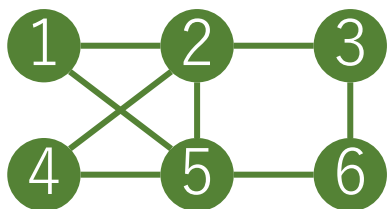
0 1 ... 9 からの文字列

与えられた正整数（十進法で書く）が素数か

問題
HAMILTON

0 1 ... 9 (), からの文字列

与えられたグラフ（次のような形式で書く）が
ハミルトン閉路をもつか
（全頂点を一度ずつ通る辿り方）



符号化

6, (1,2), (1,5), (2,3), (2,4),
(2,5), (3,6), (4,5), (5,6)

※ それなりに常識的な符号化法なら計算量の議論に支障ないことが多いので
今後あまり明記しないこともある



機械は文字列で表せる

チューリング機械は (有限の) 文字列で記述できる
遷移規則 $\delta(q, \sigma) = (q', \sigma', d)$ が有限個あるだけ

その文字列 (算譜) ^{プログラム} を機械と呼ぶと思ってもよい

万能機械



プログラマ

このお蔭で 各機械を実際に建造せずとも
算譜を読み込むことで同じ機能を実現できる

問題
EVAL

入力
答

二つの文字列の組 (M, x)
機械 M は入力 x を受理するか

「算譜を実行する」問題

定理
EVAL $\in$ CE

EVAL を認識するチューリング機械を
「万能チューリング機械」という

証明 (?) 「チャーチとチューリングの定立」に頼ると……

先程やったように
チューリング機械 (を表す図) を見ながらそれを実行することは
機械的作業である (ので同じことをチューリング機械でできる)

ϵ
0
1
00
01
10
11
000
001
010
011
100
101
110
111
0000
0001
0010
0011
0100
0101

すべての機械は
このどこかに現れる

認識可能でない言語

対角線論法



文字列に対応
づけて無限に

○と×を横に並べたものを○×行と呼び
○×行を縦に並べたものを○×表と呼ぶ

定理

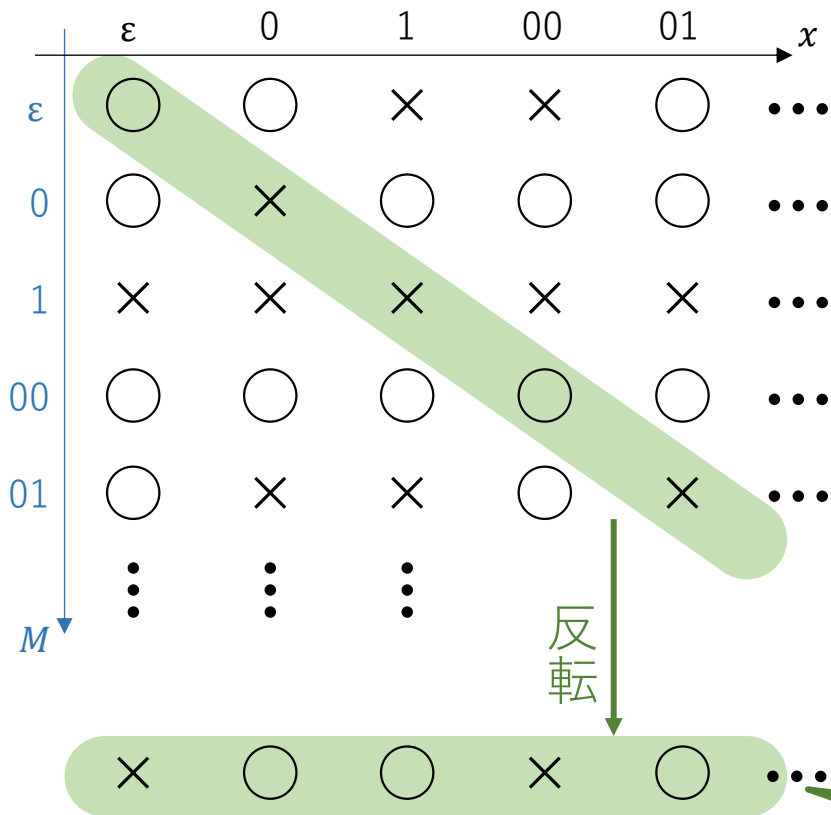
どんな $\bigcirc \times$ 表に対しても
それに現れない $\bigcirc \times$ 行が存在する

∴ 左図のように 対角線上の内容と
喰い違うように定義すればよい

各行を機械の動作と考えると……

チューリング機械 M が入力 x を受理するか否かを第 (M, x) 成分に $\bigcirc \times$ で記した表にこの定理を適用

どの機械でも認識されない言語





今の議論で結局 何という問題が認識不能と示されたのか

問題 EVAL

入力 二つの文字列の組 (M, x)

答 機械 M は入力 x を受理するか

認識可能

問題 $\overline{\text{EVAL}}$
(EVAL の補集合)

入力 二つの文字列の組 (M, x)

答 機械 M は入力 x を受理しないか

認識可能でない

対角線論法で
作った問題

入力 文字列 x

答 機械 x は入力 x を受理しないか

認識可能でない

EVAL は認識可能 (CE に属する) だが
判定可能 (自身も補言語も認識可能) ではない

任意の入力 x に対し

- もし $x \in \text{EVAL}$ ならば x を受理 (して停止)
- もし $x \notin \text{EVAL}$ ならば x を受理せずに停止すること

M に x を入力した計算が
停止しないことを
有限の時間で確実に
予言する方法はない



問題
SR

入力 書換え規則の (有限) 集合 R と文字列 $w \in \Sigma^*$

R の各規則は $u \rightarrow v$ という形 ($u, v \in \Sigma^*$)

文字列の一部を u から v に書換えることができるという意味

答 R による書換えを次々と w に施して^{空文字列} ε にできるか

つまり
 $xuy \Rightarrow_R xvy$
とできる

$w \Rightarrow_R^* \varepsilon$ と
書くこと
にする

例 1

入力

$$R \begin{cases} aa \rightarrow bbb \\ aba \rightarrow a \\ bb \rightarrow \varepsilon \end{cases}$$

$$w = aababab$$

答 ○ (受理)

$$\left(\begin{array}{l} aababab \Rightarrow_R aabab \Rightarrow_R aab \Rightarrow_R \\ bbbb \Rightarrow_R bb \Rightarrow_R \varepsilon \text{ とできるので} \end{array} \right)$$

例 2

入力

$$R \begin{cases} aa \rightarrow bab \\ aba \rightarrow a \\ bb \rightarrow \varepsilon \end{cases}$$

$$w = aababab$$

答 × (不受理)

$$\left(\boxed{\text{?}} \text{ ので} \right)$$



問題
SR

入力 書換え規則の (有限) 集合 R と文字列 $w \in \Sigma^*$

R の各規則は $u \rightarrow v$ という形 ($u, v \in \Sigma^*$)

文字列の一部を u から v に書換えることができるという意味

つまり
 $xuy \Rightarrow_R xvy$
とできる

答 R による書換えを次々と w に施して^{空文字列} ε にできるか

$w \Rightarrow_R^* \varepsilon$ と
書くこと
にする

例 1

入力

$R \begin{cases} aa \rightarrow bbb \\ aba \rightarrow a \\ bb \rightarrow \varepsilon \end{cases}$

$w = aababab$

答

○ (受理)

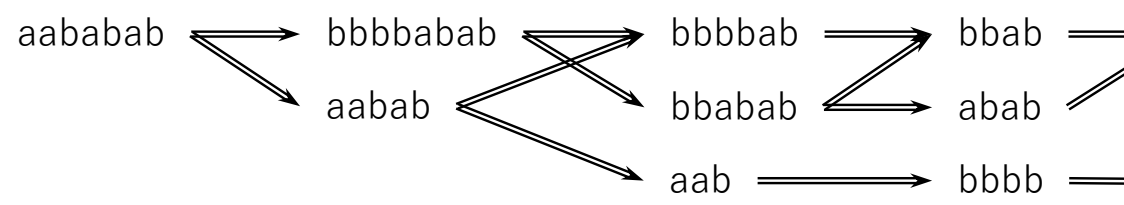
$\left(\begin{array}{l} aababab \Rightarrow_R aabab \Rightarrow_R aab \Rightarrow_R \\ bbbb \Rightarrow_R bb \Rightarrow_R \varepsilon \text{ とできるので} \end{array} \right)$

定理
SR $\in$ CE

書換え 1 回で作れる文字列をすべて列挙 (ε があれば受理)

書換え 2 回で作れる文字列をすべて列挙 (ε があれば受理)

……と順に調べてゆけばよい



(この方法では $w \Rightarrow_R^* \varepsilon$ でないときは計算が停止しない)



実は SR は判定可能でないことが後で判る

その前にまず 二つの似た問題のどちらが
より判定可能でありそうか比べる論法 (帰着) について考えよう

問題

SR

入力

書換え規則の集合 R と文字列 w

答

R による書換えを次々と w に施して空文字列にできるか

問題

SR'

入力

書換え規則の集合 R と文字列 w, w'

答

R による書換えを次々と w に施して w' にできるか

どちらが
より難しい?



問題の難しさの比較 (帰着)

これが解ければ

こっちも解ける



入力

そのためには

$(R, w) \dots\dots\dots (R, w, \varepsilon)$

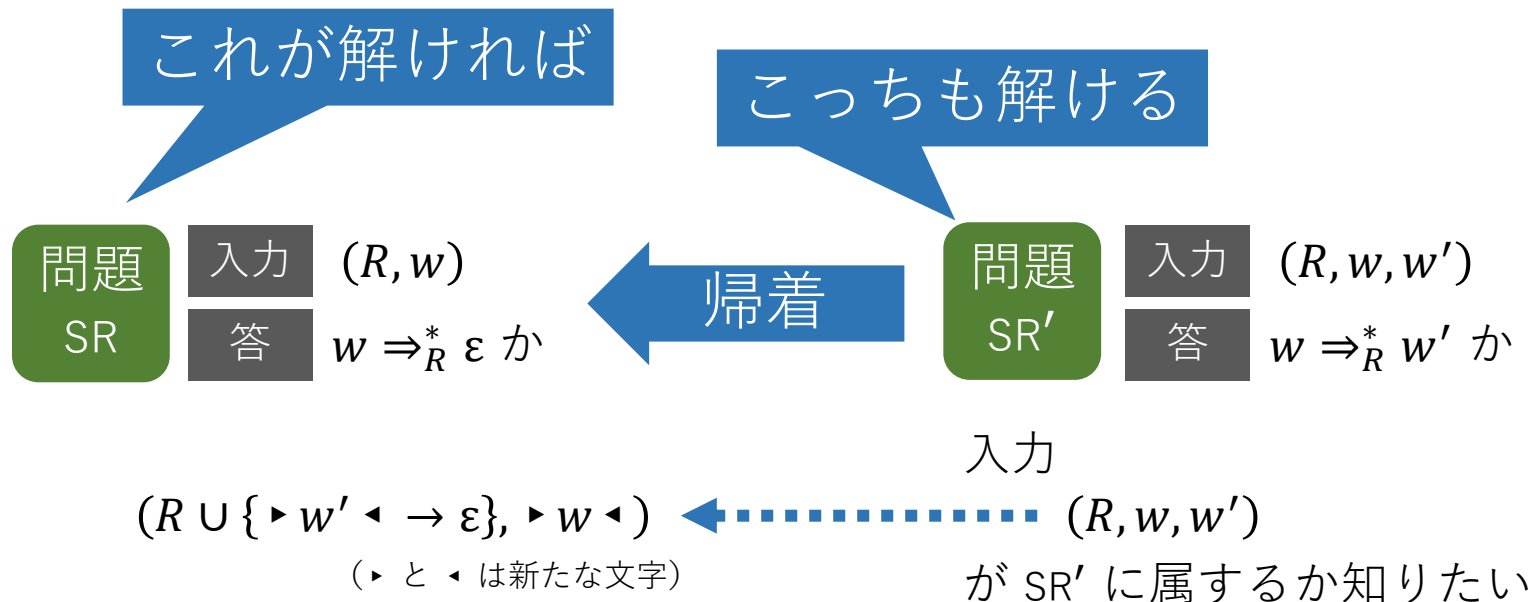
が SR に属するか知りたい

が SR' に属するか調べれば良い

$$(R, w) \in \text{SR} \iff (R, w, \varepsilon) \in \text{SR}'$$



問題の難しさの比較 (帰着)

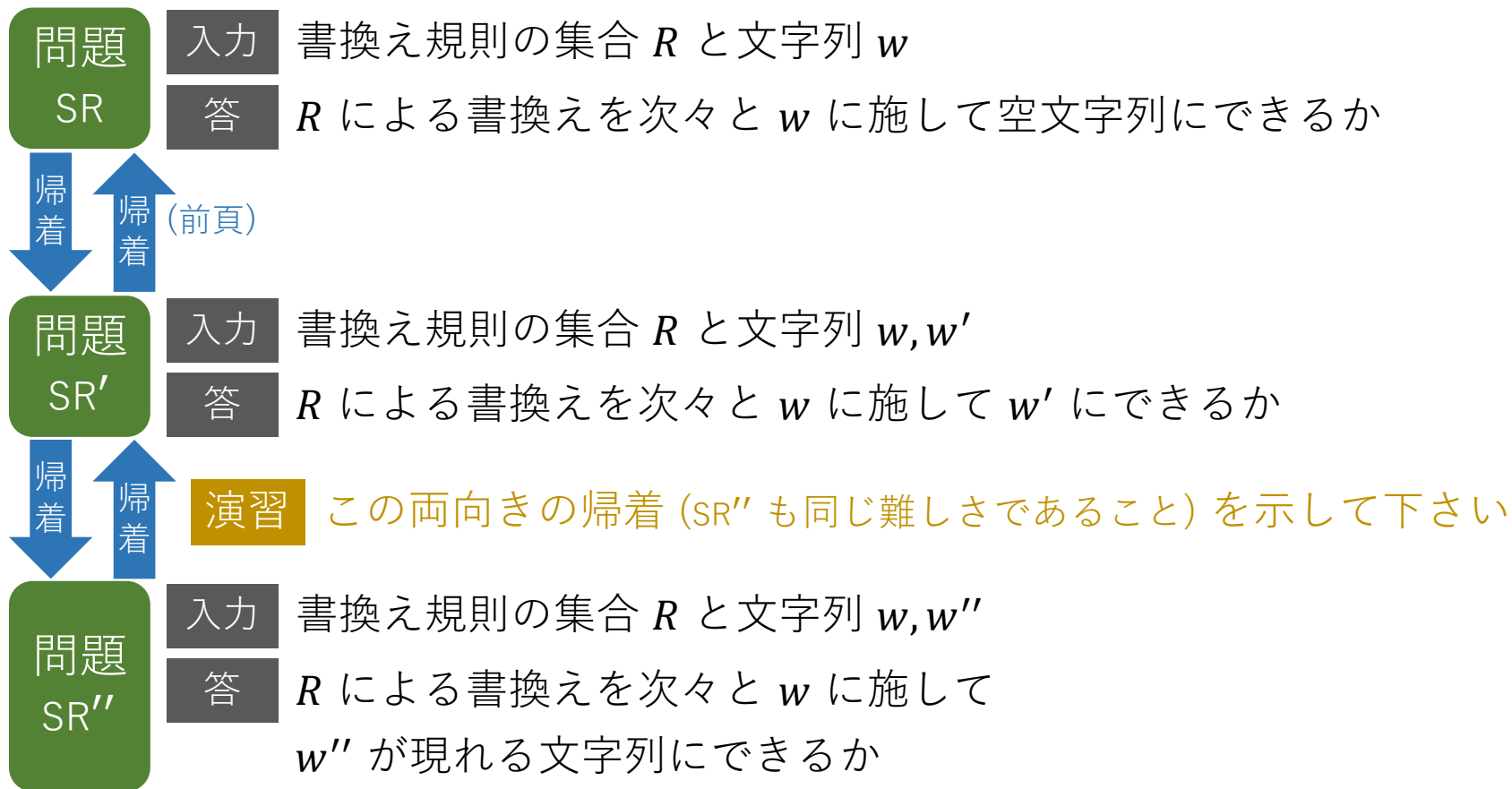


$$(R \cup \{\triangleright w' \blacktriangleleft \rightarrow \varepsilon\}, \triangleright w \blacktriangleleft) \in \text{SR} \iff (R, w, w') \in \text{SR}'$$

二つの問題は (判定可能かどうかに関しては)
「同じ難しさ」であることが判った！



問題の難しさの比較 (帰着)





定理

SR は判定不能 (すなわち $\overline{SR} \notin CL$)

これが判定不能と判っているので

これも

これも判定不能

∴ 右図の帰着による



$$(M, x) \dashrightarrow (R, \triangleright q_{\text{始}} x \triangleleft, \textcircled{\text{受}})$$

但し R は M の各状態 $q \in Q$ と文字 $\sigma \in \Gamma$ について次の規則を加えて作る

- もし $\delta(q, \sigma) = (q', \sigma', \textcircled{\text{左}})$ ならば 任意の $\tau \in \Gamma$ に対し規則 $\tau q \sigma \rightarrow q' \tau \sigma'$
- もし $\delta(q, \sigma) = (q', \sigma', \textcircled{\text{右}})$ ならば 規則 $q \sigma \rightarrow \sigma' q'$
- もし $\delta(q, \sigma) = \text{止}$ かつ $q \in Q_{\text{受理}}$ ならば 規則 $q \sigma \rightarrow \textcircled{\text{受}}$

また 規則 $\triangleright \rightarrow \triangleright \sqcup$ および規則 $\triangleleft \rightarrow \sqcup \triangleleft$ も R に含める すると

$$\begin{aligned} (M, x) \in \text{EVAL} &\Leftrightarrow \text{状況 } q_{\text{始}} x \text{ から遷移 } \xrightarrow{M} \text{ を辿ってゆくと} \\ &\quad \delta(q, \sigma) = \text{止} \text{ かつ } q \in Q_{\text{受理}} \text{ なる } q \sigma \text{ が現れる状況に到達} \\ &\Leftrightarrow \text{文字列 } \triangleright q_{\text{始}} x \triangleleft \text{ に } R \text{ の規則を施して} \left(\begin{array}{l} \text{すなわち} \\ (R, \triangleright q_{\text{始}} x \triangleleft, \textcircled{\text{受}}) \in \text{SR''} \end{array} \right) \\ &\quad \textcircled{\text{受}} \text{ を含む文字列に辿り着ける \end{aligned}$$



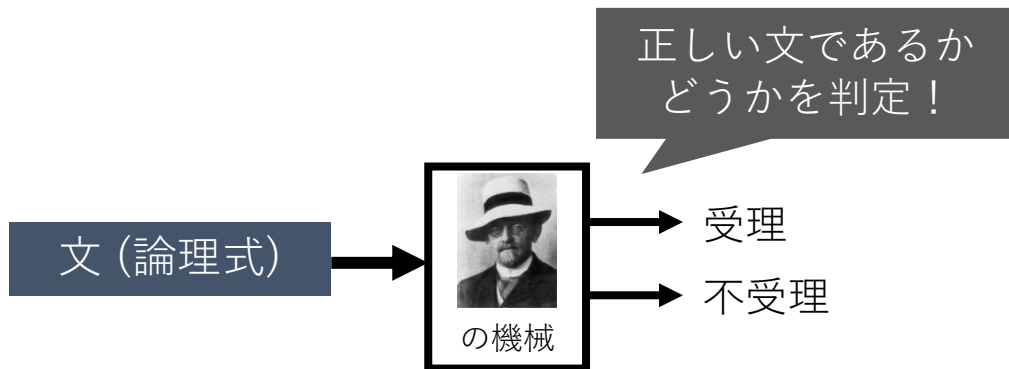
ヒルベルトの判定問題



一階述語論理で書かれた文が与えられたとき
それが正しい (= 証明可能) か否かを判定する
ような機械的な手順はあるか？ (1928)



A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* 2, **42**: 230–265, 1936.

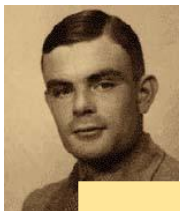




ヒルベルトの判定問題

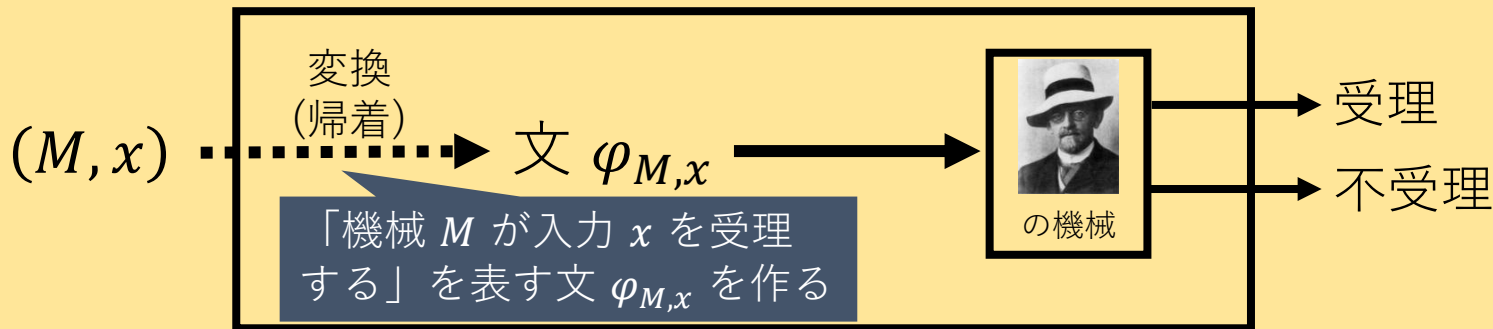


一階述語論理で書かれた文が与えられたとき
それが正しい (= 証明可能) か否かを判定する
ような機械的な手順はあるか? (1928)



A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* 42: 230–265, 1936.

もしあるとすると先程の EVAL が判定できてしまうので、無い



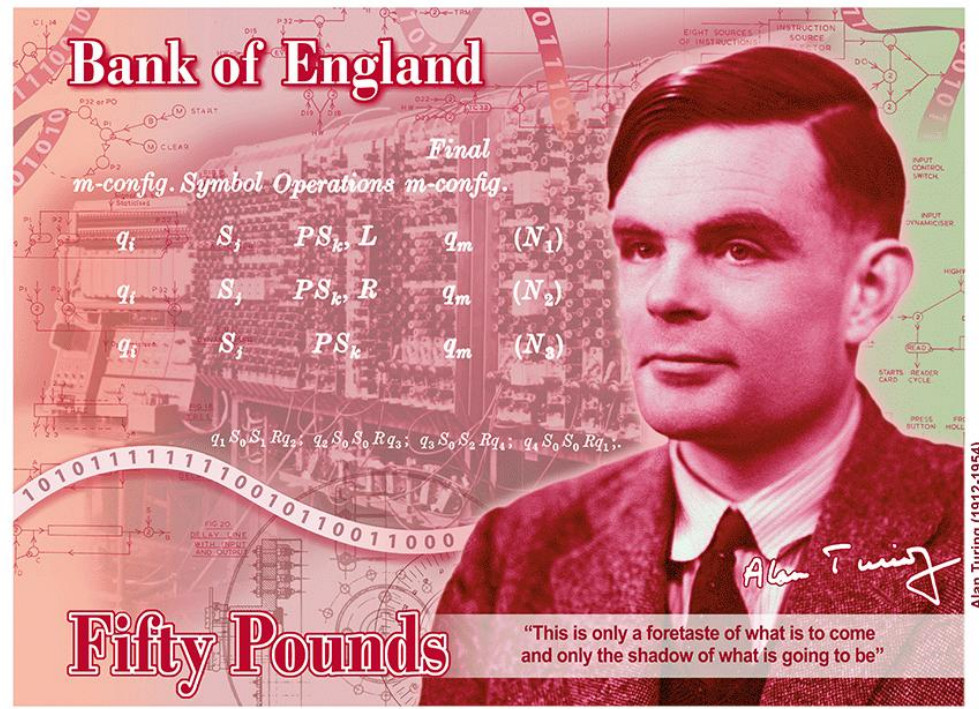


五十ポンド紙幣（2021年から）



BANK OF ENGLAND

Alan Turing Banknote Concept



©The Governor and Company of the Bank of England 2019



まとめ 第一日 計算可能性

- 問題 = 言語 無限個の入力に正解を定めたもの
- 機械 = 算法 有限に記述される計算手順
- チューリング機械 (書込み機能あり) により認識される
= 認識可能 (**CE**) $\doteq$ 「計算できる」
- 正則 (書込み機能なし) でないが認識可能である言語が存在する
- 機械は有限の文字列 (算譜) で記述できる
- 入力された算譜を実行する計算ができる ($\text{EVAL} \in \mathbf{CE}$)
- しかし $\overline{\text{EVAL}} \notin \mathbf{CE}$ 特に EVAL は判定不能 (対角線論法)
- そこから様々な言語の判定不能性が帰着により示される