

現代の数学と数理解析

計算能力の階層

令和4年5月13日 河村

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



1. 問題と機械

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	×	○	○	×	○	×	○	…

a と b からなる

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 7 とか ε ^{空文字列} とか 119 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

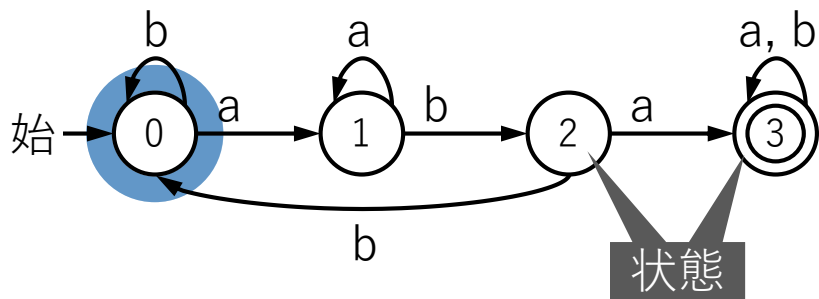
この講義では「文字列を入力すると それが PRIME に属するか
答えてくれる計算機械」などについて考えたい

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考える**) 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



状態は有限個
矢印も有限本

有限の記述で書き表せる

a b b a a b a b



受理

a b b a a b b a



不受理

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

右端まで終わったとき状態が $Q_{\text{受理}}$ 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

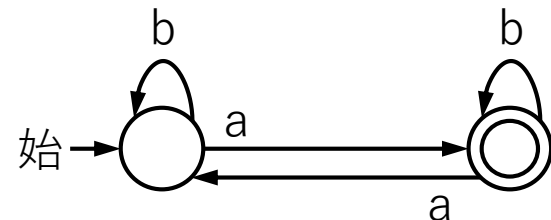
すべての入力で
正しく判断!

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	...

問1 a と b からなる文字列のうち

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

をそれぞれ認識する有限状態機械を作ってください

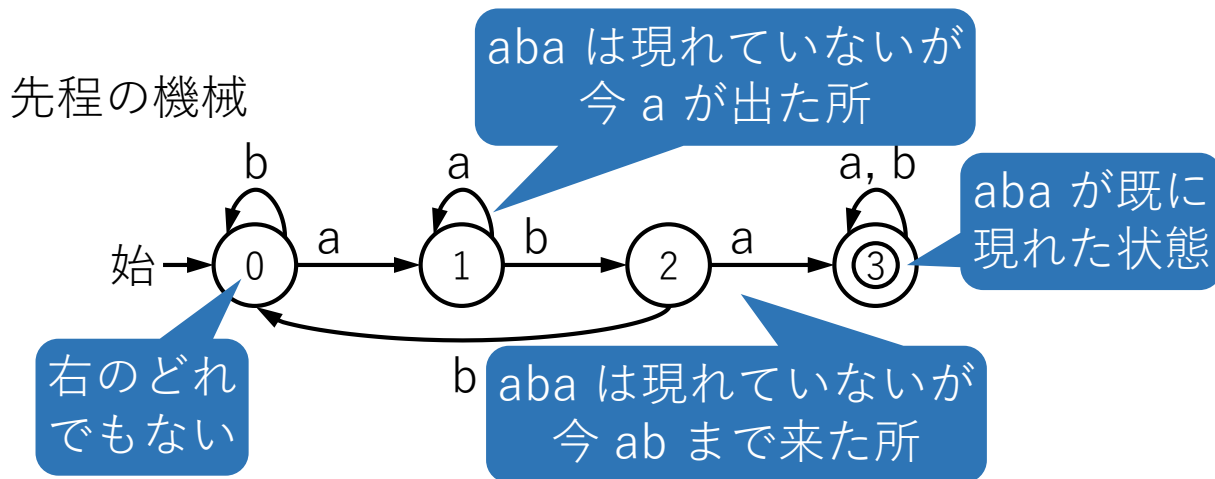


(1) の答 (例)

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

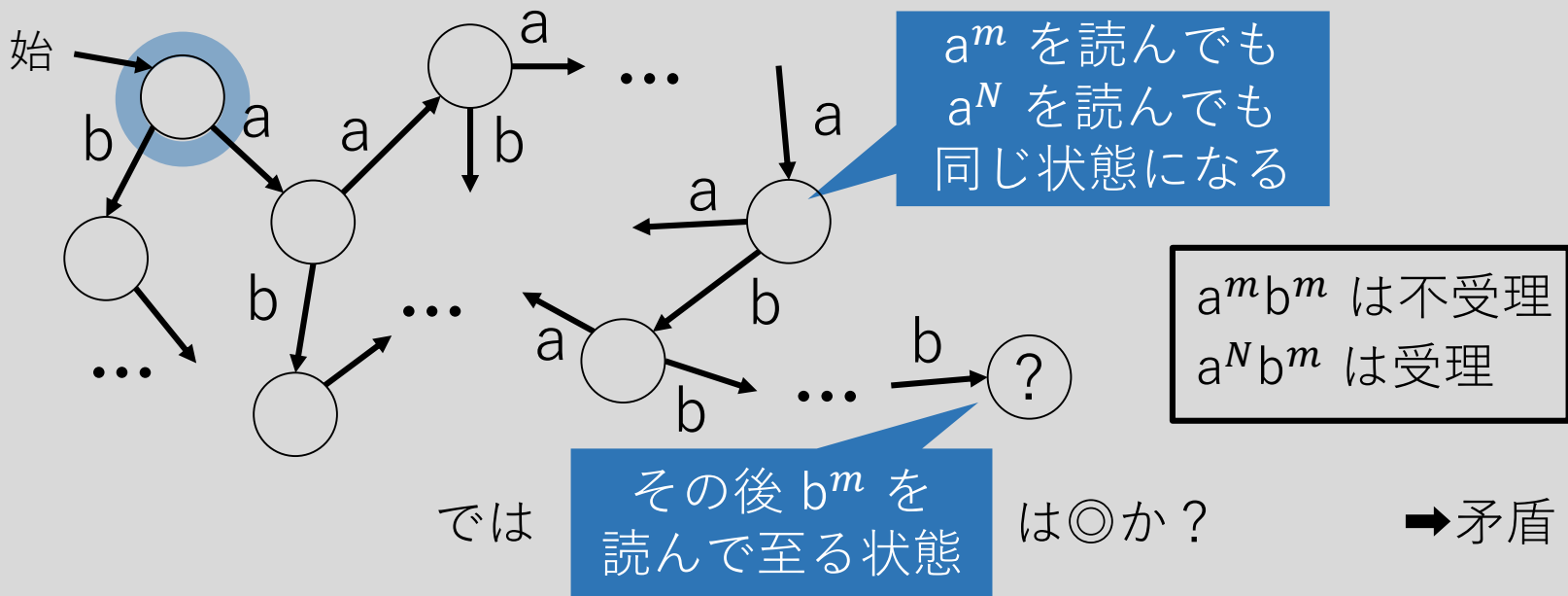
与えられた文字列に
a が b よりも多く
現れるか答えよ

問2 同じ文字列を二度繰り返したものの全体
つまり言語 $\{ww \mid w \in \{a,b\}^*\}$ を認識する
有限状態機械が存在しないことを示せ
※このスライドのような絵ではなく きちんとした文章で書くこと

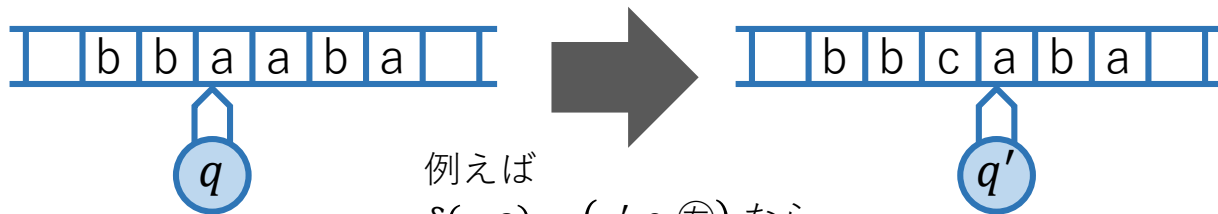
定理

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？



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

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** $\delta: Q \times \Gamma \rightarrow (Q \times \Gamma \times \{\text{Ⓢ}, \text{Ⓣ}\}) \cup \{\text{止}\}$

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

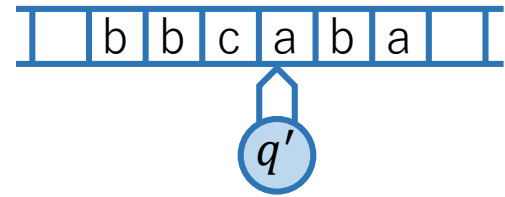
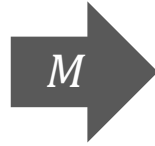
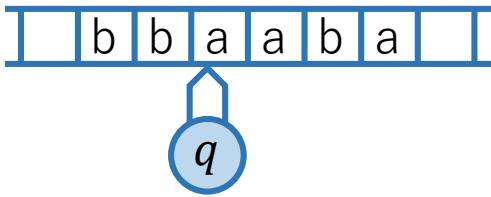
状態 $q \in Q$ で文字 σ を読むと $\delta(q, \sigma)$ が

- **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



状況の遷移

この状況を bbqaaba
のように表すことにする

例えば $\delta(q, a) = (q', c, \oplus)$ 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 \square を加えた文字列も同じ状況とみなす) 状況の遷移 M を次で定義する

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

- もし $\delta(q, \sigma) = (q', \sigma', \oplus)$ 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 $\delta(q, \sigma) = (q', \sigma', \ominus)$ 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

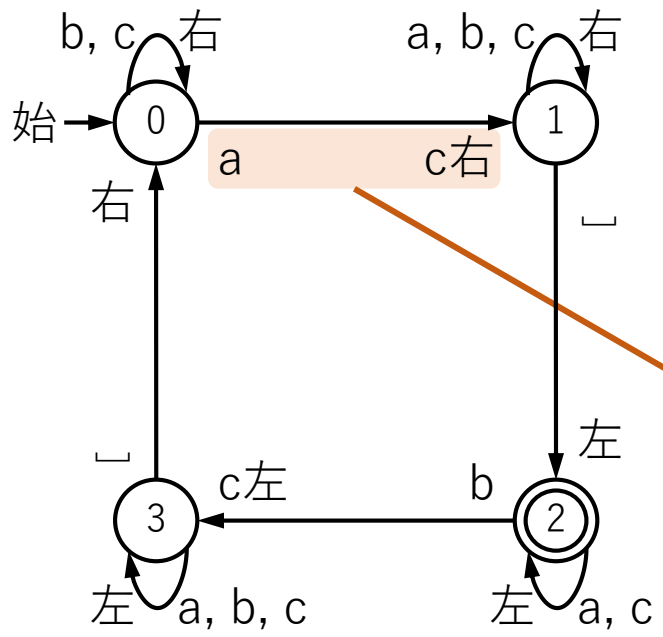
状況の有限列 (s_0, \dots, s_f) であって次を満すものが存在することをいう

- $s_0 = q_{\text{始}}x$
- $s_0 \xrightarrow{M} s_1 \xrightarrow{M} s_2 \xrightarrow{M} \dots \xrightarrow{M} s_f$
- 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

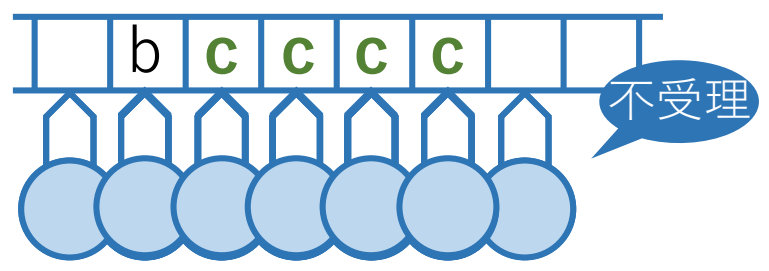
MOREA を認識する
チューリング機械



- 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

A state transition diagram with two states, 0 and 1. State 0 is the start state. Transitions from 0: 'b, c' to the right stays in 0; 'a' to the right goes to 1. Transitions from 1: 'a, b, c' to the right stays in 1. A light orange box highlights the transition from 0 to 1 on 'a'.

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

A state transition diagram with two states, 2 and 3. State 2 is the start state. Transitions from 2: 'b' to the left goes to 3; 'a, c' to the left stays in 2. Transitions from 3: 'c' to the left goes to 2; 'a, b, c' to the left stays in 3.

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

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**であるという

• ABA

正則

• MOREA

認識可能

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 (a)]. 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
考えてよかろう

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 999999999999999 are the same.

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.

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

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



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

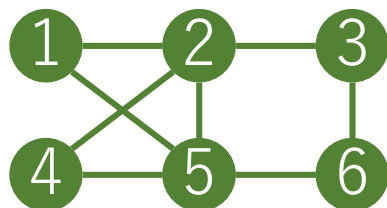
0 1 ... 9 からの文字列

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

1. 問題と機械 まとめ

- 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
- 有限状態機械 (書込み機能なし) により認識される = 正則
- 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 \doteq 「計算できる」
- 正則でないが認識可能である言語が存在する

2. 機械の万能性と限界

※ 以下「チューリング機械」を単に機械という

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（^{プログラム}算譜）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 は認識可能

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筈だ）

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

しかし 機械が有限の文字列で表せることから

認識可能でない言語

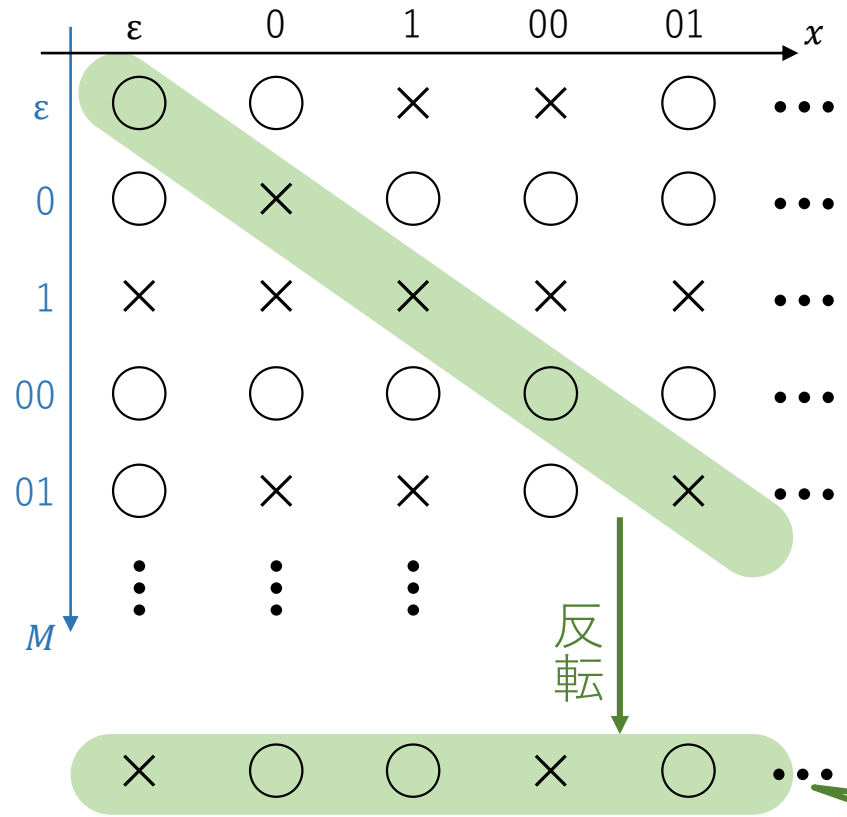
を構成することもできてしまう

対角線論法

文字列に対応
づけて無限に

図のように

○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○×表に対しても
それに現れない○×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○×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 は認識可能だが**判定**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 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)

$$\left(a \text{ を消せる規則がないので} \right)$$

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

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

例 1

入力

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

$w = aababab$

答

○ (受理)

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

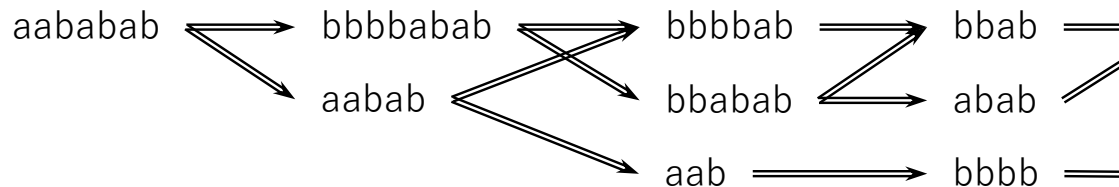
定理

SR は認識可能

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 $w \Rightarrow_R^* \epsilon$ 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

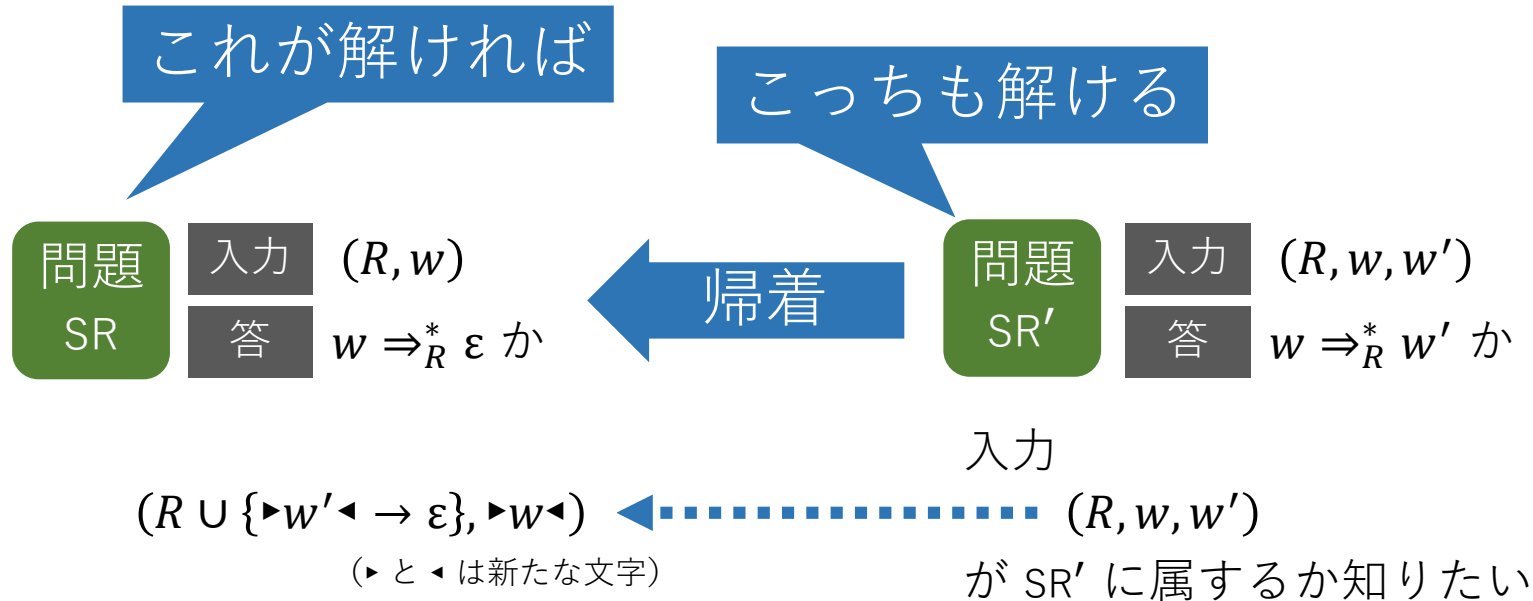
$(R, w) \dots\dots\dots (R, w, \varepsilon)$

が SR に属するか知りたい

が SR' に属するか調べれば良い

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

問題の難しさの比較 (帰着)

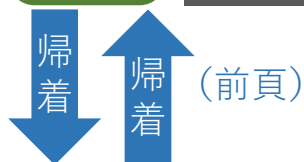


$$(R \cup \{\triangleright w' \blacktriangleleft \rightarrow \varepsilon\}, \triangleright w \blacktriangleleft) \in SR \iff (R, w, w') \in SR'$$

二つの問題は (決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)

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



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'' が現れる文字列にできるか

定理

SR は判定可能でない (SR は認識できない)

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



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

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

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

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

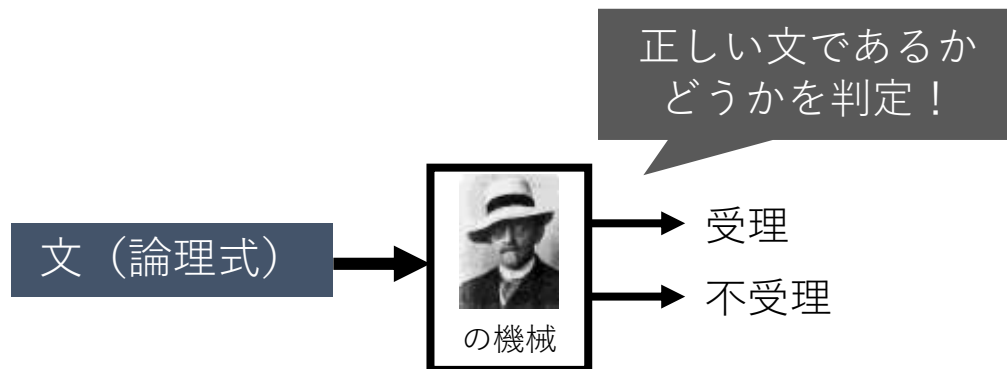
ヒルベルトの判定問題



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

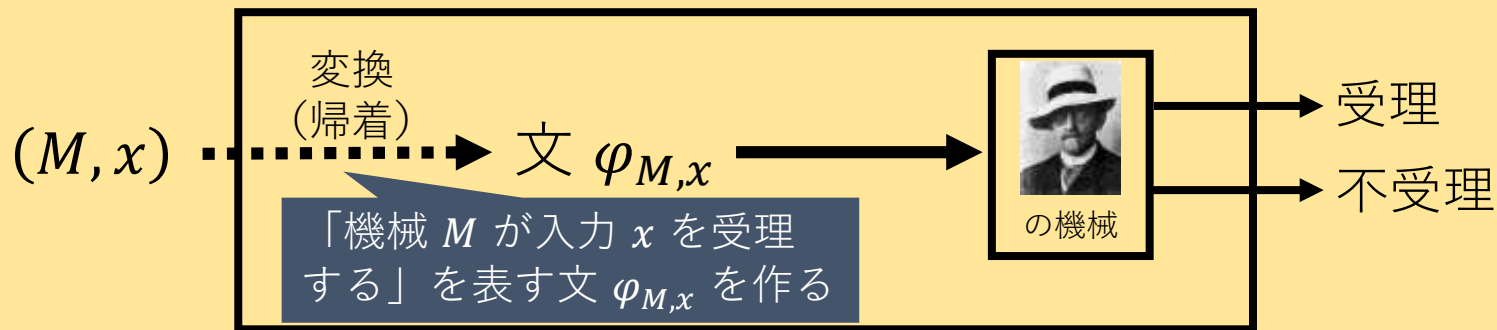


一階述語論理で書かれた文が与えられたときそれが正しい (= 証明可能) か否かを判定するような機械的な手順はあるか? (1928)



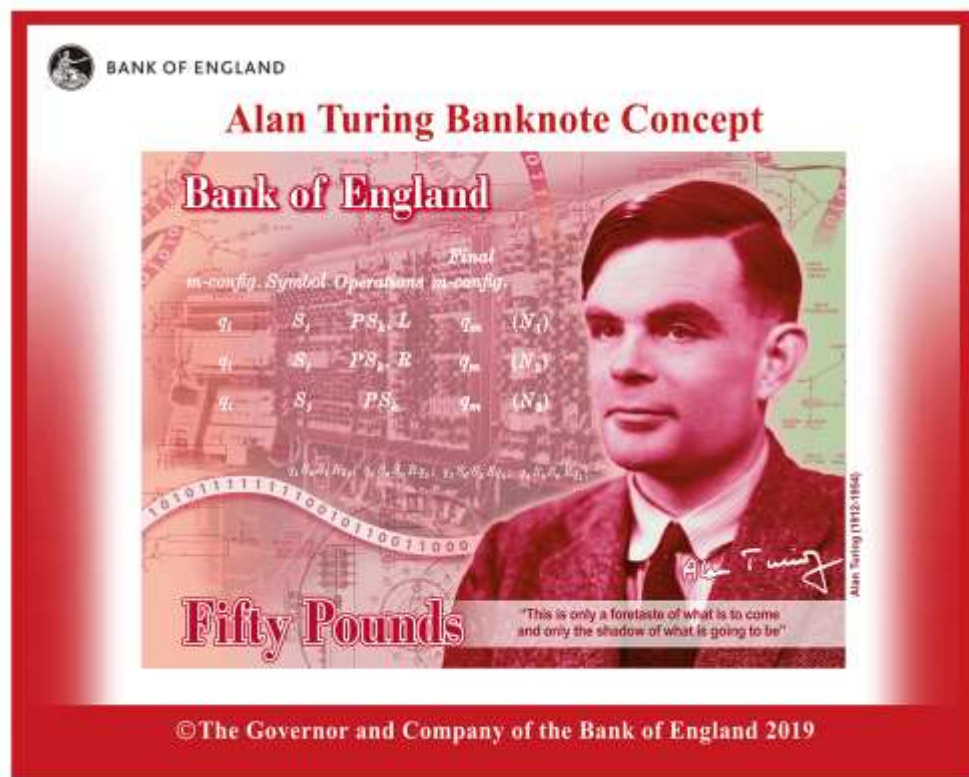
A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* 42: 230–265, 1936.

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



2. 機械の万能性と限界 まとめ

- 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（EVAL は認識可能）
- しかし EVAL は判定可能ではない（対角線論法）
- 様々な言語の判定不能性が帰着により示される



英国五十ポンド紙幣（二〇二一年）

3. 計算量

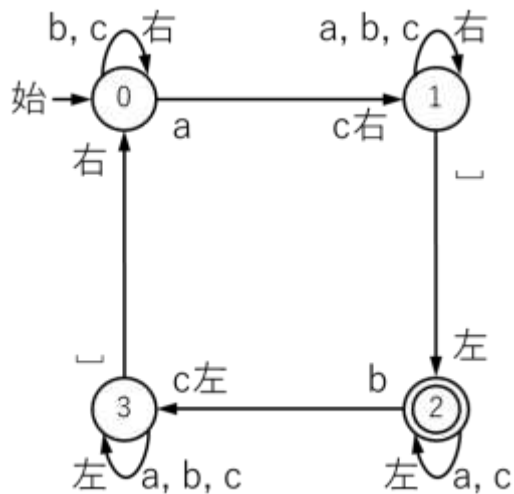
計算量の考え方（原則） 1950～70年代

- チューリング機械での計算にかかる時間（遷移の回数）や空間（訪れる欄の数）を考える
現実にかかる時間や空間をよく表している
- それが入力の長さに応じてどう変るか関数として表す
「長さ n の入力なら必ず時間（や空間）が $T(n)$ 以内で済む」
ような関数 T が計算量の尺度
- その関数の増大の速さに着目
特に n の多項式以内か否かが重要

定義

機械 M が**多項式時間**であるとは 或る多項式 p が存在し
 どの長さ n の入力に対しても M の計算が
 時間 $p(n)$ 以内に終る (遷移 $p(n)$ 回以下で停止する) ことをいう
 言語 A を認識する多項式時間の機械 M が存在するとき
 A は**多項式時間認識可能**であるという

例 先程のこの機械は
 時間 $(n+1)^2$ 以内に停止するので
 MOREA は多項式時間認識可能



n の多項式以内
 である例

n
 $n\sqrt{n}$
 $n^2 \log n$
 $5n^3 + 4n$

n の多項式以内
 でない例

2^n
 2^{2^n}
 1.05^n
 $n \log n$
 $n!$

問4

MOREA を認識する もっと速い 機械を作れ

$\lim_{n \rightarrow \infty} \frac{T(n)}{n^2} = 0$ なる時間 $T(n)$ 以内に終る

定義

機械 M が**多項式時間**であるとは 或る多項式 p が存在し
どの長さ n の入力に対しても M の計算が
時間 $p(n)$ 以内に終る (遷移 $p(n)$ 回以下で停止する) ことをいう
言語 A を認識する多項式時間の機械 M が存在するとき
 A は**多項式時間認識可能**であるという

「認識可能」の定義では $x \notin A$ のときは停止しないことを許していた
それに合せて「多項式時間認識」の定義は 長さ n の入力 x について

- $x \in A$ のとき M は x を時間 $p(n)$ 以内に受理する
- $x \notin A$ のとき M は x を時間 $p(n)$ 以内に受理しないとすべきでは？

そう定義しても「多項式時間
認識可能」の意味は変わりません

受理せずに時間 $p(n)$ が経過したら
不受理と確定できるので

多項式時間では「認識可能」と「判定可能」の違いはない
(多項式時間認識可能な言語は補集合も多項式時間認識可能)



チャーチとチューリングの定立（続）

「現実的な手間で計算できる」 = (チューリング機械で) 多項式時間認識可能

チューリング機械の定義の細部は
計算法が多項式時間であるかないかに
影響を与えない

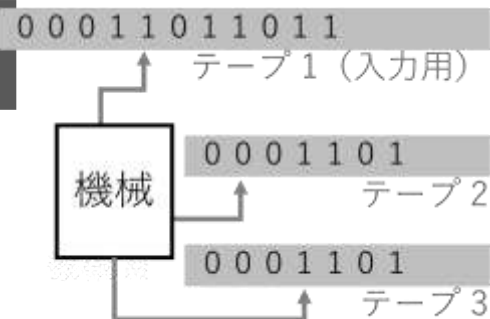
「 $O(n^3)$ であるかないか」など細かい量には
それなりに影響がある

今後は一々チューリング機械を考えず 計算手順が解り易いように説明する

時間 = 「基本的な操作の回数」と大雑把に考えてよい

- 四則演算やビットの操作など
- 機械語での命令数

テープの形や
読取り部の動き方など



なぜ多項式時間か否かが重要か

■ 入力が大きくなると手間が大違い

入力長 $n =$	10	30	50	100	1000	1万	100万	1億
\sqrt{n}	1秒以内	1秒以内	1秒以内	1秒以内	1秒以内	1秒以内	1秒以内	1秒以内
n	1秒以内	1秒以内	1秒以内	1秒以内	1秒以内	1秒以内	1秒	2分
n^2	1秒以内	1秒以内	1秒以内	1秒以内	1秒	2分	12日	3百年
n^3	1秒以内	1秒以内	1秒以内	1秒	17分	12日	3万年	3百億年
2^n	1秒以内	18分	36年	4京年	1秒に100万回の処理ができるとしたときにかかる時間			
10^n	17分	3京年						
$n!$	3.6秒	800京年						

多項式時間
↑
指数時間
↓

実際には 多項式時間の中の速さの差 (n^2 と n^3 の違いなど) も勿論重要なのですが この講義では「多項式時間であるかないか」というより大きな違いに着目します



なぜ多項式時間か否かが重要か

■ 指数個（以上）の組合せから何かを探す場面は多い（組合せ爆発）

問題
PRIME

与えられた正整数（十進法で n 桁）が素数か判定

それ未満の数（ 10^n 個ほどある）で割り切れるか全部調べれば判る

それよりも劇的に速く n の多項式時間で判定する方法が存在 [AKS04]

問題
HAMILTON

与えられたグラフがハミルトン閉路をもつか判定
(全頂点を一度ずつ通る辿り方)

頂点の並べ方（ $n!$ 個ある）を全部調べれば判る

n の多項式時間で判定する方法があるかどうかは判らない（多分なさそう）

[AKS05] M. Agrawal, N. Kayal and N. Saxena. PRIMES is in P. *Annals of Mathematics*, 160: 781–793, 2004.

定義

機械 M が**多項式空間**であるとは 或る多項式 p が存在し
どの長さ n の入力に対しても M の計算が
空間 $p(n)$ 以内である (テープ上で初めの欄の左右 $p(n)$ 個の範囲に留まる) ことをいう
言語 A を認識する多項式空間の機械 M が存在するとき
 A は**多項式空間認識可能**であるという

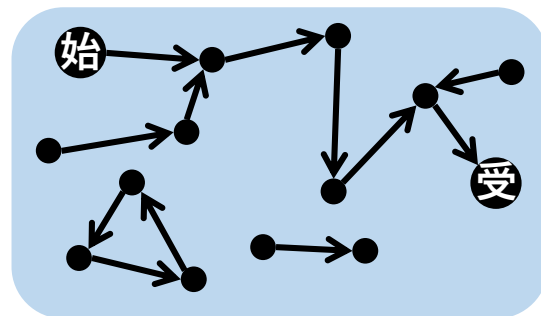
多項式 q が存在して
入力長 n のとき $2^{q(n)}$ 時間以内

定理

多項式時間認識可能 \Rightarrow ① 多項式空間認識可能 \Rightarrow ② 指数時間認識可能

- ① 空間を 1 使う (新たな欄に移動する) にも
時間 1 かかるので
- ② 多項式空間機械 M に対して多項式 q が存在し
長さ n の入力に対する M の状況として
あり得るものは $2^{q(n)}$ 個以内

受理するのであれば時間 $\leq 2^{q(n)}$ で受理する



状況間の遷移関係

※ PRIME は本当は多項式時間認識可能であることが今では判っているが

問題
PRIME

入力

正の整数 X (を十進表示したもの)

答

X は素数か

問題
HAMILTON

入力

グラフ G

答

G にハミルトン閉路はあるか

定理

PRIME や HAMILTON は多項式空間認識可能

n 桁の割り算ひとつひとつは容易 (n の多項式時間)

入力

48581583386419 ÷
← n 桁 →

2 = 24290791693209 あまり 1
3 = 16193861128806 あまり 1
⋮
6306457 = 7703467 あまり 0
48581583386418

↓
前の行を覚えておく必要はないが
（同じ場所に書きやすい）
指数的に多くの手間がかかるが

※ PRIME は本当は多項式時間認識可能
であることが今では判っているが

問題
PRIME

入力

正の整数 X (を十進表示したもの)

答

X は素数か

問題
HAMILTON

入力

グラフ G

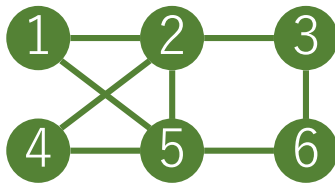
答

G にハミルトン閉路はあるか

定理

PRIME や HAMILTON は多項式空間認識可能

入力



n 頂点

辺がちゃんと繋がった経路に
なっているか確かめるのは容易

- 1 2 3 4 5 6 } ダメ (3と4の間や6と1の間が繋がっていないので)
- 1 2 3 4 6 5 } ダメ (3と4の間や4と6の間が繋がっていないので)
- 1 2 3 5 4 6
- ⋮
- 6 5 4 3 2 1 } 経路の候補 $n!$ 個



指数的に多くの手間がかかるが
前の行を覚えておく必要はない
(同じ場所に上書きしてよい)

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 ε にできるか

問題

SR \geq

入力 書換え規則の集合 R と文字列 w

但し R の各規則 $u \rightarrow v$ において $(u \text{ の長さ}) \geq (v \text{ の長さ})$

答 R による書換えを次々と w に施して ε にできるか

問題

SR \geq^1

入力 書換え規則の集合 R と文字列 w

但し R の各規則 $u \rightarrow v$ において $(u \text{ の長さ}) \geq (v \text{ の長さ})$

答 R による書換えを次々と w に施すと
可能な書換え方は毎回一通りしかなく やがて ε に達する

問題

SR \geq^1

入力 書換え規則の集合 R と文字列 w

但し R の各規則 $u \rightarrow v$ において $(u \text{ の長さ}) > (v \text{ の長さ})$

答 R による書換えを次々と w に施すと
可能な書換え方は毎回一通りしかなく やがて ε に達する

困難そう
(一般的な入力)

容易そう
(特殊な入力)

問題

SR

入力 (R, w)

答 $w \Rightarrow_R^* \varepsilon$ か

問題

SR_≥

SR を各規則が
(旧長さ) ≥ (新長さ)
の場合に制限したもの

問題

SR_≥¹

SR_≥ を更に
可能な書換え方が毎回一通り
な場合に制限したもの

問題

SR_>¹

SR_≥¹ を更に各規則が
(旧長さ) > (新長さ)
の場合に制限したもの

定理 (再)

SR は認識可能 (だが
判定可能でない)

? (次頁)

定理

SR_≥¹ は多項式空間認識可能

定理

SR_>¹ は多項式時間認識可能

書換えを
実際に順次
(機械の
テープ上で)
行ってみればよい

定理

SR_{\geq} は多項式空間認識可能

※ $\Sigma = \{a, b\}$ の場合を考える (他のときも同様)

問題
 SR_{\geq}

入力 (R, w) 但し各規則が
(旧長さ) \geq (新長さ)

問 $w \Rightarrow_R^* \varepsilon$ か

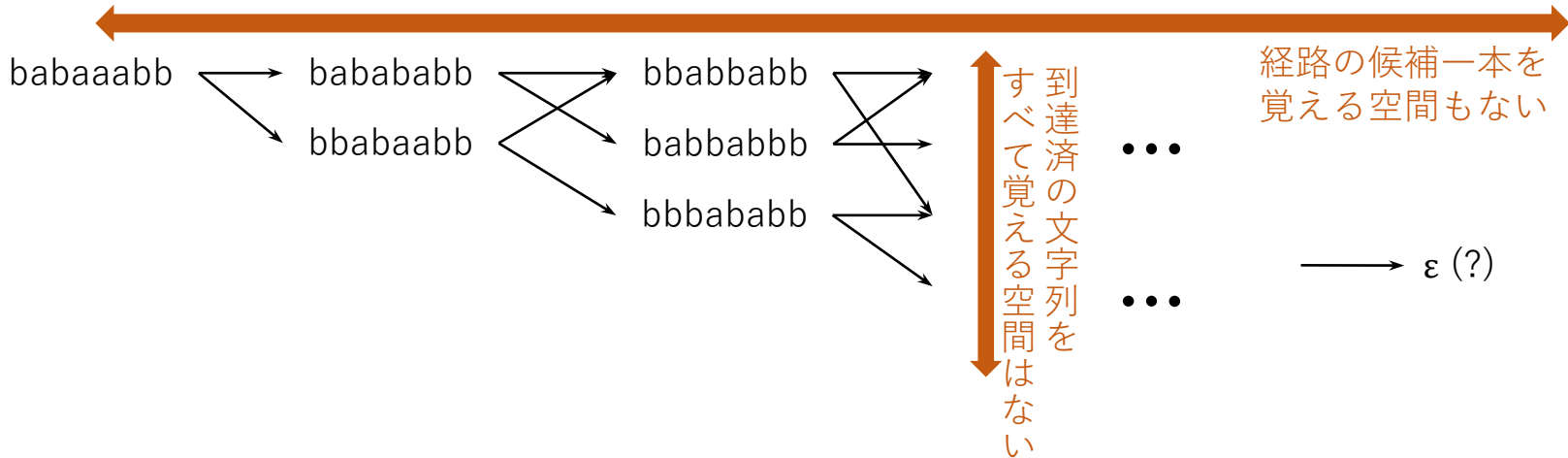
入力の長さが n である (w の長さ $< n$) とき

w から書換えにより生じ得る文字列も長さ $< n$ であり その個数は $< 2^n$

したがって $w \Rightarrow_R^{\leq 2^n} \varepsilon$ かどうか調べればよい

書換え 2^n 回以内で w から ε が得られる という意味

しかし 単純に長さ $\leq 2^n$ の経路すべてを調べることはできない



定理

SR_{\geq} は多項式空間認識可能

問題
 SR_{\geq}

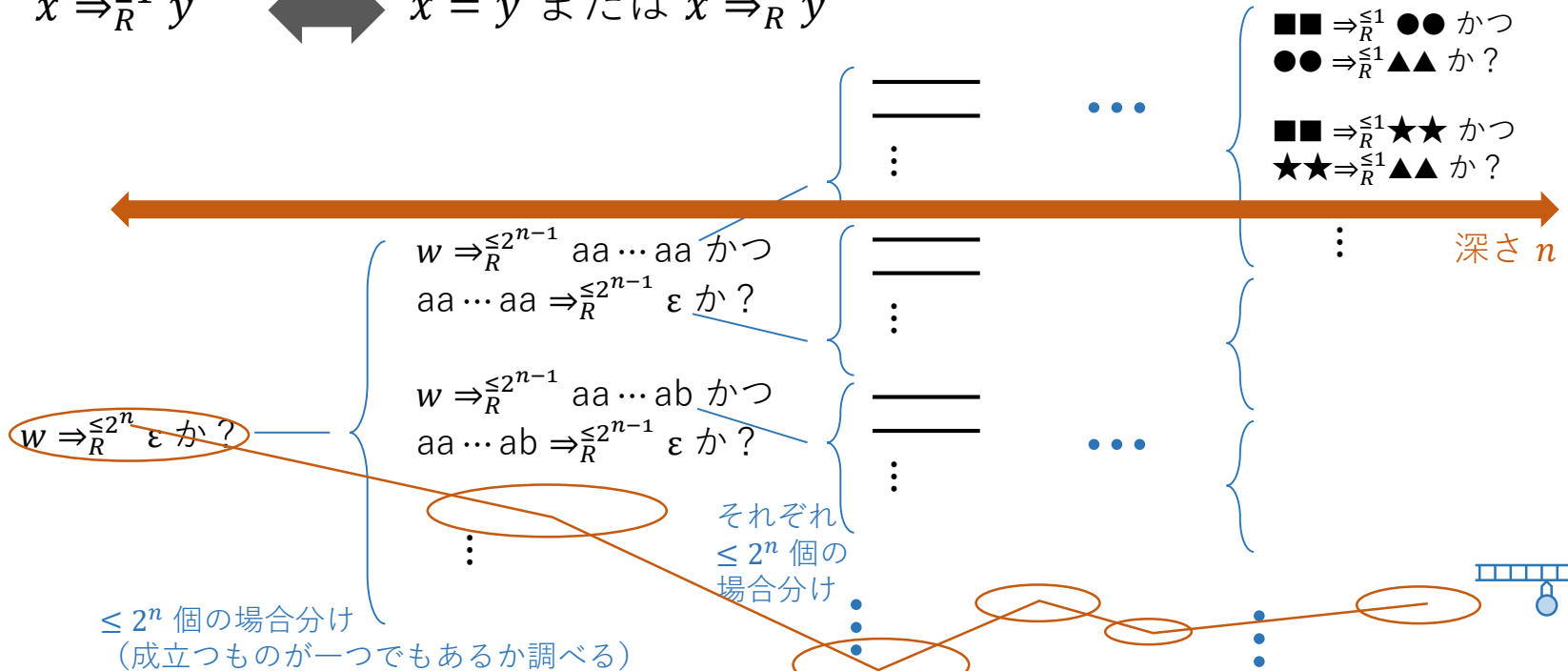
入力 (R, w) 但し各規則が
(旧長さ) \geq (新長さ)

問 $w \Rightarrow_R^* \varepsilon$ か

$w \Rightarrow_R^{\leq 2^n} \varepsilon$ かどうか 次の関係を再帰的に用いて調べればよい

$x \Rightarrow_R^{\leq 2^{i+1}} y$ \iff 或る $z \in \Sigma^{<n}$ が存在して $x \Rightarrow_R^{\leq 2^i} z$ かつ $z \Rightarrow_R^{\leq 2^i} y$

$x \Rightarrow_R^{\leq 1} y$ \iff $x = y$ または $x \Rightarrow_R y$



複雑さの階層 問題の難しさの分類

多項式時間認識可能でなく
多項式空間認識可能である
問題が存在するかは未解決

多項式空間認識可能でなく
指数時間認識可能である
問題が存在するかも未解決

但し 多項式時間認識可能でなく
指数時間認識可能である問題が
存在することは判っている
(対角線論法の応用)

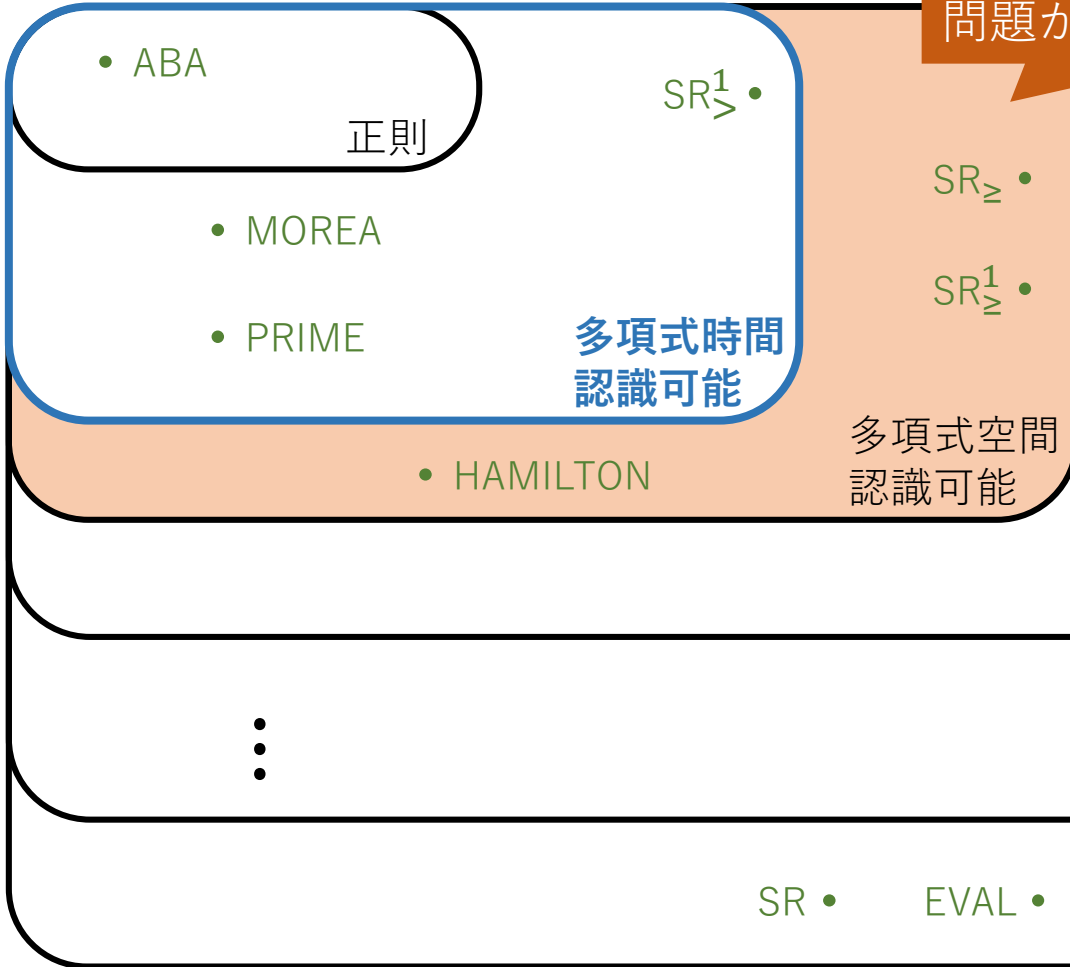
指数時間
認識可能

...

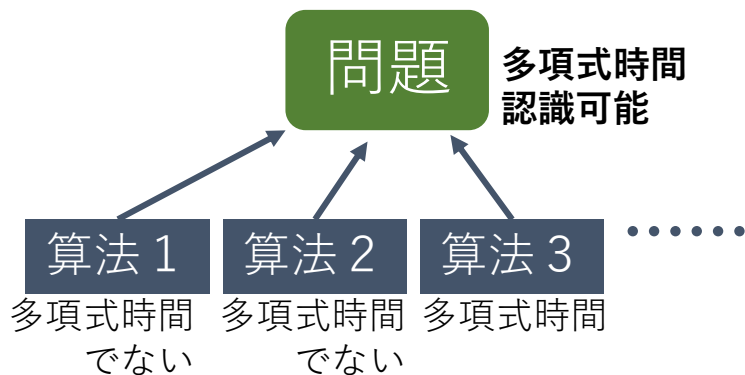
判定可能
(時間制限なし)

SR • EVAL •

認識可能



問題の難しさを証明するのは難しい



問題□□は容易に解けるか？



効率の良い算法 (= 機械) が 存在するか？

「問題□□は如何なる算法を以てしても (時間△△では) 解けない」
という証明をすることは難しく なかなか成功していない……

定義

言語 A が言語 B に**多項式時間帰着**するとは
多項式時間機械 T が存在して
任意の入力 x に対する T の出力 $T(x)$ が
 $x \in A \Leftrightarrow T(x) \in B$ を満すことをいう

停止したときの
テープ上の文字列

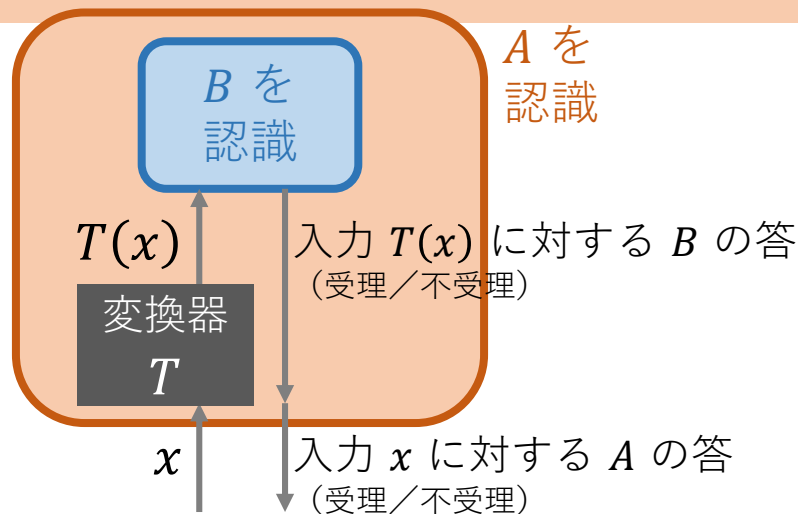
$A \leq^P B$ と書く

向きに注意

「 A の難しさは B 以下」という
気持ちなのでこの不等号で表す

「 B を使うと A が計算できる」

「 B が多項式時間計算可能ならば A も然り」を成立たすには
必ずしも問題 A の入力 x を問題 B の入力 $T(x)$ に対応させるという上記の形でなくてもよい
(例えば $x \in A$ かどうか知るために幾つかの文字列 y_1, y_2, \dots が B に属するか否かを使うとか)
つまり上記の定義は「帰着」のうち特殊な形のものである
しかし今日の話にはこの形の帰着で十分なので 以下ではこれを単に「帰着」という



$A \leq^P B$ であるとする...

もし B が多項式時間認識可能ならば A も多項式時間認識可能

もし B が多項式空間認識可能ならば A も多項式空間認識可能

もし B が判定可能ならば A も判定可能

もし B が認識可能ならば A も認識可能

問題どうしの**相対的な困難さの比較**ができる

問題

入力

(M, x)

EVAL

答

M は x を受理するか

問題

入力

(R, w)

SR

答

$w \Rightarrow_R^* \varepsilon$ か

先程は $EVAL \leq^P SR$ を示すことで

EVAL の判定不能性から SR の判定不能性を導いた

(その時点では帰着が多項式時間どうかは気にしていなかったが)

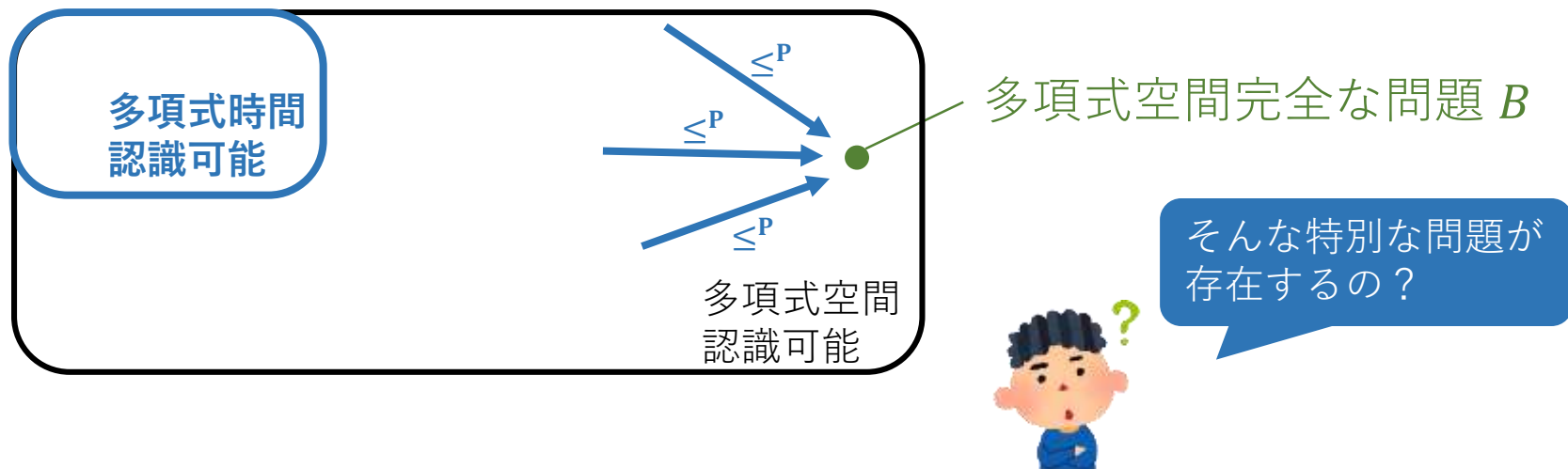
定義

多項式空間認識可能な言語 B が**多項式空間完全**であるとは
任意の多項式空間認識可能な言語 A が B に多項式時間帰着する
($A \leq^P B$) ことをいう

「 B は多項式空間認識可能な問題のうち最難」

「多項式時間認識可能 = 多項式空間認識可能でない限り
 B は多項式時間で解けない」

「多項式空間認識可能とは 複雑さが問題 B 以下であること」



問題 EVAL

入力 文字列の組 (M, x)

答 機械 M は入力 x を受理するか

問題 EVAL_{space}

入力 文字列の組 $(M, x, 0^s)$

答 機械 M は入力 x を空間 s 以下で受理するか

定理

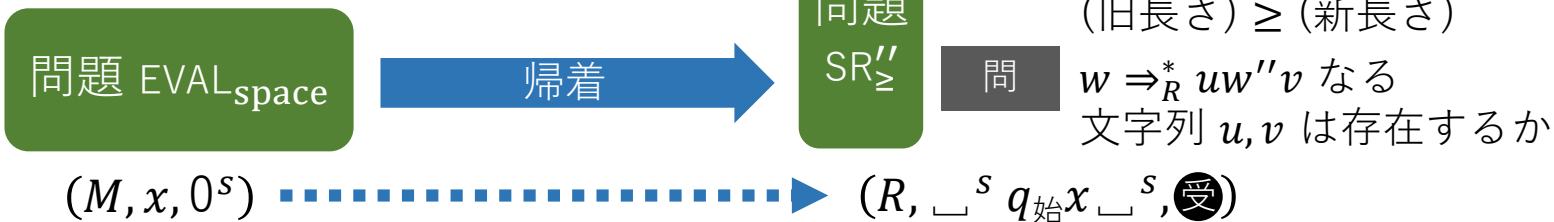
EVAL_{space} は多項式空間完全

- ∴ 多項式空間認識可能な言語 A を考える
- A を認識する多項式空間の機械 M が存在する
- すなわち多項式 p が存在し
- 任意の長さ n の入力 x に対して M は空間 $\leq p(n)$ で停止する
- そこで $T(x) = (M, x, 0^{p(n)})$ とすると
- T は A から EVAL_{space} への帰着

定理

SR_{\geq} は多項式空間完全 (同じ証明により SR_{\geq}^1 も)

先程の $EVAL \leq^P SR'' \leq^P SR$ と同様に
 $EVAL_{space} \leq^P SR''_{\geq} \leq^P SR_{\geq}$ を示す



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 $\delta(q, \sigma) = (q', \sigma', \text{左})$ ならば 各 $\tau \in \Gamma$ に対し規則 $\tau q \sigma \rightarrow q' \tau \sigma'$
- もし $\delta(q, \sigma) = (q', \sigma', \text{右})$ 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 $q \sigma \rightarrow \text{受}$

($EVAL \leq^P SR''$ のときに設けた「空白文字を端に付け足す規則」 $\triangleright \rightarrow \triangleright \sqcup$ および $\triangleleft \rightarrow \sqcup \triangleleft$ は設けない)

すると M は x を空間 $\leq s$ で受理 $\iff R$ の規則に従って $\sqcup^s q_{始} x \sqcup^s$ から 受 を含む文字列に到達できる が成立

様々な分野の多項式空間完全問題

様々な到達可能性を判定する問題（文字列の書換え以外にも）

二人ゲームの勝負を判定する問題（囲碁など） [LS80]

不確実さの下での最適化問題 [Pap85]

有限状態機械の操作に関する問題 [Koz77, JR93]

周期的なスケジューリングや配置計画を最適化する問題 [Orl81]

常微分方程式の数値計算に関する問題 [Kaw10]

[LS80] D. Lichtenstein and M. Sipser. Go is polynomial-space hard. *Journal of the ACM*, **27**: 393–401, 1980.

[Pap85] C. H. Papadimitriou. Games against nature. *Journal of Computer and System Sciences*, **31**: 288–301, 1985.

[Koz77] D. Kozen. Lower bounds for natural proof systems. In *Proc. 18th Symposium on the Foundations of Computer Science*, 254–266, 1977.

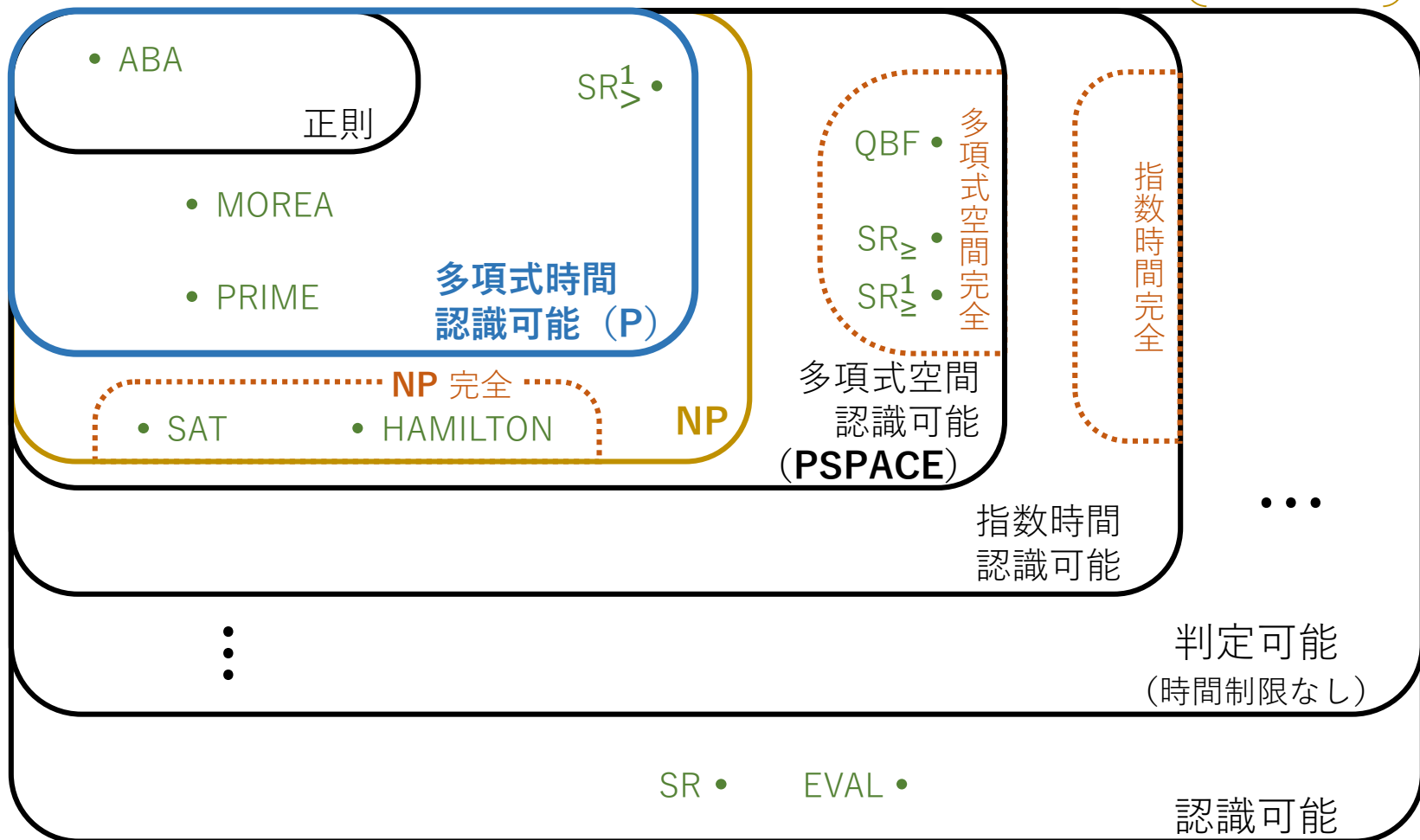
[JR93] T. Jiang and B. Ravikumar. Minimal NFA problems are hard. *SIAM Journal on Computing*, **22**: 1117–1141, 1993.

[Orl81] J. B. Orlin. The complexity of dynamic languages and dynamic optimization problems. In *Proc. 13th ACM Symposium on the Theory of Computing*, 218–227, 1981.

[Kaw10] A. Kawamura. Lipschitz continuous ordinary differential equations are polynomial-space complete. *Computational Complexity*, **19**: 305–332, 2010.

複雑さの階層と完全問題

NP については
本講義では
触れなかった



3. 計算量 まとめ

- 時間計算量は「入力長 n のとき時間 $T(n)$ 以内で計算できる」という形で測る（空間計算量も同様）
- **多項式時間** \doteq 現実的な手間での計算
- **多項式空間** は指数個の（非現実的に多くの）調べ尽しができる
- しかし「多項式時間認識可能 \neq 多項式空間認識可能」は未証明（不可能性の証明は難しい）
- $\text{○○完全} =$ 「○○のうち最難」の問題（帰着により定義）
- 多様な ○○完全問題 見た目は違えど困難さの核心は同じ

様々な計算量級 クラス

本講義で扱った

- 正則 (REG)
- 多項式時間認識可能 (P)
- 多項式空間認識可能 (PSPACE)
- 判定可能 (R または Decidable)
- 認識可能 (RE または CE)

以外にも問題の複雑さの度合を色々な側面から捉えた基準が数多く定義され研究されている

ウェブサイト「Complexity Zoo」 500以上の計算量級を紹介
https://complexityzoo.net/Complexity_Zoo



Complexity Zoo

Introduction

Welcome to the **Complexity Zoo**... There are now 545 cl

Complexity classes by letter: [Symbols](#) - [A](#) - [B](#) - [C](#) - [D](#) - [E](#) -

P

[P](#) - [P/log](#) - [P/poly](#) - [P^{#P}](#) - [P^{#P\[1\]}](#) - [P_{CTC}](#) - [PAC⁰](#) - [PBP](#) - [k-PB](#)
[- PH](#) - [PH^{CC}](#) - [Φ₂P](#) - [PhP](#) - [Π₂P](#) - [PINC](#) - [PIO](#) - [P^K](#) - [PKC](#) - [PI](#)
[polyL](#) - [PostBPP](#) - [PostBPP^{CC}](#) - [PostBQP](#) - [PP](#) - [PP^{CC}](#) - [PP/_f](#)
[PromiseBPP](#) - [PromiseBQP](#) - [PromiseP](#) - [PromiseRP](#) - [Pro](#)
[PT/WK\(f\(n\),g\(n\)\)](#) - [PZK](#)

Q

[Q](#) - [QAC⁰](#) - [QAC⁰\[m\]](#) - [QACC⁰](#) - [QAC_f⁰](#) - [QAM](#) - [QCFL](#) - [QC](#)
[QMIP](#) - [QMIP_{1e}](#) - [QMIP_{ne}](#) - [QNC](#) - [QNC⁰](#) - [QNC_f⁰](#) - [QNC¹](#) - [Q](#)

R

[R](#) - [RBQP](#) - [RE](#) - [REG](#) - [RevSPACE\(f\(n\)\)](#) - [RG](#) - [RG\[1\]](#) - [R_H](#)

クラス

様々な計算量級

「Complexity Zoology」

<https://www.math.ucdavis.edu/~greg/zoology/>

指数時間

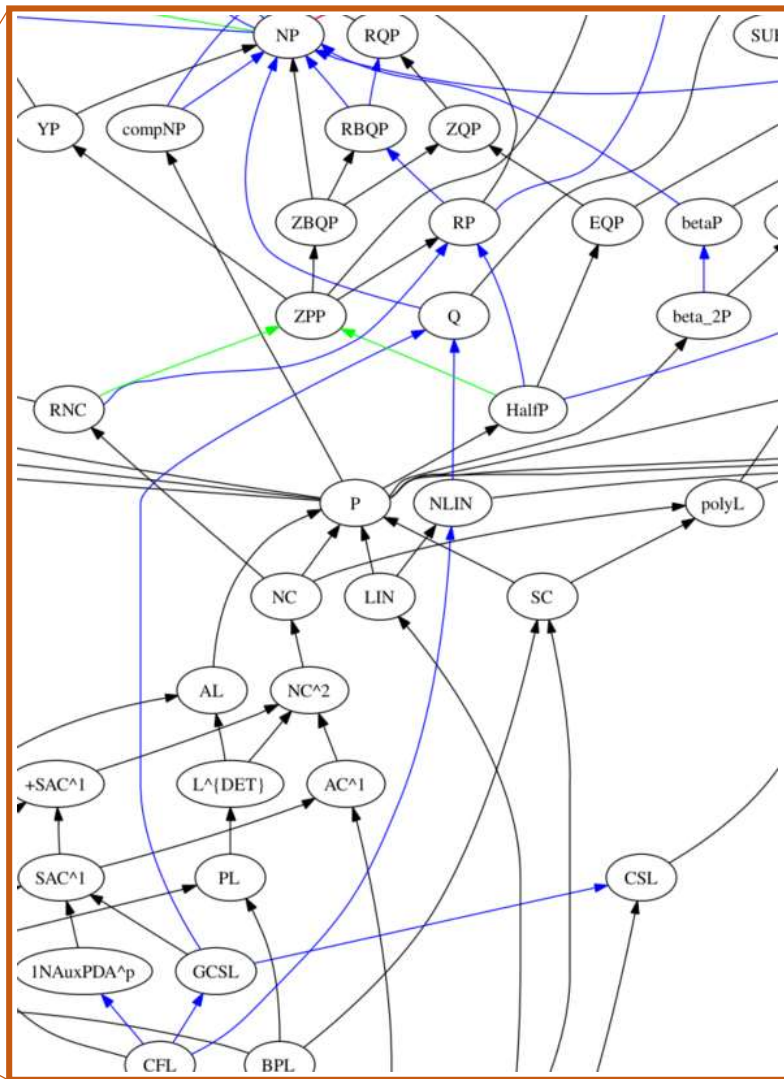
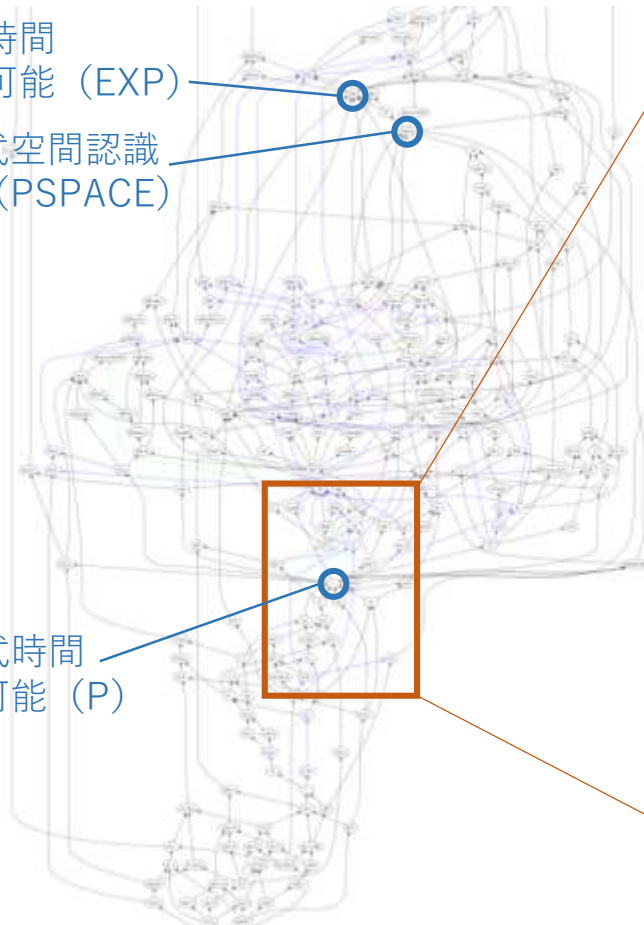
認識可能 (EXP)

多項式空間認識

可能 (PSPACE)

多項式時間

認識可能 (P)



M・シプサ著、太田・田中監訳『計算理論の基礎 原著第二版』共立出版（平成20年）

岩間一雄『アルゴリズム理論入門』朝倉書店（平成26年）



課題

スライド中の **問1** **問2** **問3** **問4** のうち 1問以上を解いて提出せよ

提出期限 5月27日（金） 23:55

採点基準

できてる ➡ 4点

おしい ➡ 3点

できてない ➡ 0～2点

すごい ➡ 5点

特に解りやすく記述されている
独自の鋭い考察がある など