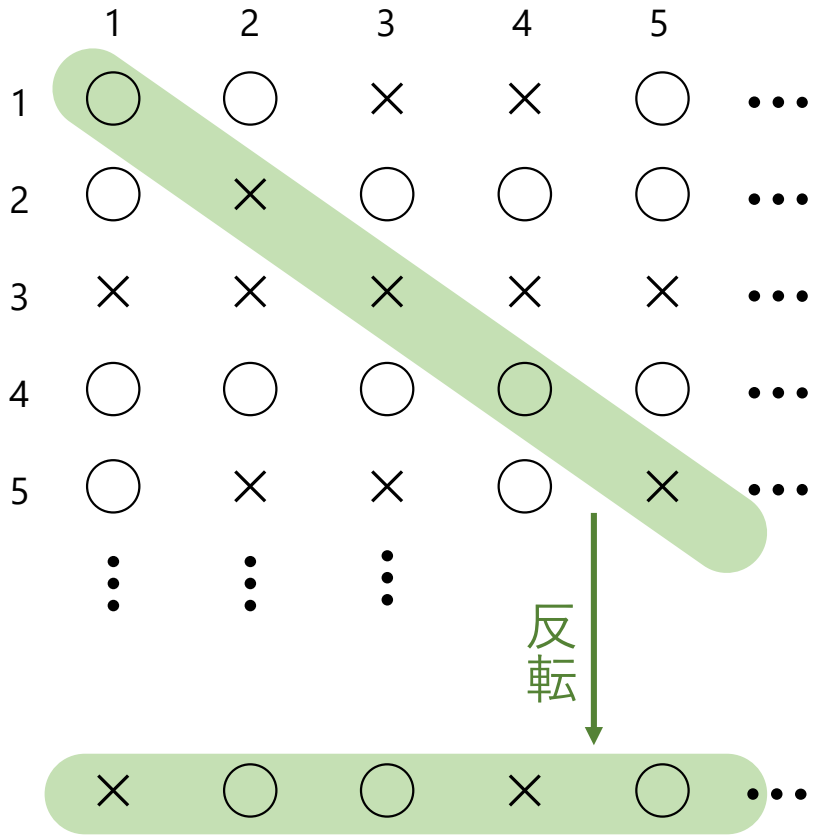
※このスライドは後で講義ページにも載ります

計算の理論

令和2年5月12日
河村彰星
（京都大学数理解析研究所）
※本日のみ担当します

16:55頃に開始します　それまで以下のことを考えてみて下さい（理由も）



左図のように
○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

番号 1 2 3 …
をつけて無限に

どちらが正しいでしょうか？

うまく○×表を作ると
あらゆる○×行が現れるようにできる

如何なる○×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

問題、算法、
チャーチ・チューリングの定立

問題とは

どの入力に対し
どの出力をすべきか定めたもの

例えば

問題 与えられた二つの正整数の最大公約数を求める

入力		出力
$(10, 8)$	➡	2
$(275, 500)$	➡	25

面倒なのでこの位の言い方にする

正確にいうと

十進法で正の整数 a を表す文字列と
十進法で正の整数 b を表す文字列とを
コンマで区切って繋げ括弧で囲んだものが入力されたとき
a と b との最大公約数を十進法で表す文字列を出力する問題
但し入力された文字列がこの形をしていなかった場合の結果は指定しない

ポスト（Emil Post）
は私の名前です

問題 ポストの文字列揃え問題
（Post's Correspondence Problem）

上下に文字列の書かれた札が何種類か与えられる
これらを横に有限枚並べて　（各種類の札が幾らでもある）
上下の文字列を一致させることができるか？

入力　　出力

b	a	ca	abc
ca	ab	a	c

➡ ハイ

（

a	b	ca	a	abc
ab	ca	a	ab	c

とできるので）

acc	ca	abc
cb	a	ab

➡ イイエ

つまり問題とは……

各入力に対し　出力すべき文字列を定めたもの
入力も出力も文字列で表される（出力はハイ／イイエのこともある）

入力	$(1,1)$	$(2,1)$	$(2,2)$	$(3,1)$	$(3,2)$	$(3,3)$	$(4,1)$	$(4,2)$	$(4,3)$	…
出力	1	1	2	1	1	3	1	2	1	…

問題とは
これ全体のこと！

問題です。275 と 500 の
最大公約数は何でしょう？

それは**入力**であって
問題ではない！

計算理論警察

アルゴリズム
算法とは

処理の手順を
きっちり定めたもの

例えば

算法 エウクレイデスの互除法
（世界最古の算法と言われる）

```
procedure euclid(a, b) {
  r := (a を b で割った余);
  if (r == 0)
    return b;
  else
    return euclid(b, r);
  fi
}
```

動作例

$euclid(275, 500)$

$275 \div 500 = 0$　余275

$euclid(500, 275)$

$500 \div 275 = 1$　余225

$euclid(275, 225)$

$275 \div 225 = 1$　余50

$euclid(225, 50)$

$225 \div 50 = 4$　余25

$euclid(50, 25)$

$50 \div 25 = 2$　余なし

25

この算法は先程の「最大公約数を求める問題」を正しく解く（証明略）

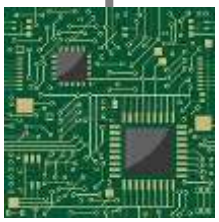
実は「ポストの文字列揃え問題」を解く算法は存在しない

しかし「存在しない」と主張するには　まず算法とは何かハッキリさせる必要がある……

チューリング機械

$x =$ 0 0 0 1 1 0 1 1 0 1 1

テープ
（右側に無限）



機械 M

内部状態は有限個
（記憶には主にテープを使う）

図のような装置であって

テープに文字列を与えて
動作させると

予め定められた規則に従って
読み書きするもの

現在の文字と
内部状態とから
次の動きが決る

遷移規則 $\delta(q, a) = (r, b, d)$

状態 q で文字 a を読むと
状態 r になり文字 b を書いて
d の向きに一マス動く

- 何らかの文字列 y を書いて停止
- 停止しない

$M(x)$ で表し
出力と呼ぶ

以下「算法」と「機械」
を全く同じ意味で使う

機械が問題を解くとは

<u>どの入力に対しても</u>
問題の要求する出力が
得られること

問題

入力 $(1,1)$ $(2,1)$ $(2,2)$ $(3,1)$ $(3,2)$ $(3,3)$ $(4,1)$ $(4,2)$ $(4,3)$ …

出力 1 1 2 1 1 3 1 2 1 …

算法

機械 M

すべてを
正しく出力

※ 問題で指定されていない入力に対しては
どうなってもよい（停止しなくても、何を出力しても）

「チューリング機械により解ける」は
定義の細部に依らない安定した重要な概念

テープの本数（入出力テープの区別、複数の作業テープ）
テープの形（両方向に無限、二次元、…）
その他の細かい規則（「左にも右にも動かない」を許すか、など）

チャーチ・チューリングの定立

数学的にハッキリした主張　　ボンヤリした主張

チューリング機械で計算可能　⇔　「計算できる」

「⇒」はまあ良さそう（チューリング機械の動きは十分単純そう）

逆に　「計算」できることはすべてチューリング機械でできるのか？

- 現実の「計算」に出て来そうな手順は確かにチューリング機械で実現できる
- 他の色々なやり方で定義された計算可能性の概念とも一致
  一般再帰函数、ラムダ計算、…

チューリングの論文

Turing, A.M. (1936). "On Computable Numbers, with an Application to the Entscheidungsproblem". *Proceedings of the London Mathematical Society* 2, **42**: 230–265.

あらゆる計算手順がチューリング機械の動きとして実現できるといえそうな理由を第9節で論じている

➡ 資料

9. *The extent of the computable numbers.*

No attempt has yet been made to show that the "computable" numbers include all numbers which would naturally be regarded as computable. All arguments which can be given are bound to be, fundamentally, appeals to intuition, and for this reason rather unsatisfactory mathematically. The real question at issue is "What are the possible processes which can be carried out in computing a number?"

The arguments which I shall use are of three kinds.

(*a*) A direct appeal to intuition.

(*b*) A proof of the equivalence of two definitions (in case the new definition has a greater intuitive appeal).

(*c*) Giving examples of large classes of numbers which are computable.

Once it is granted that computable numbers are all "computable", several other propositions of the same character follow. In particular, it follows that, if there is a general process for determining whether a formula of the Hilbert function calculus is provable, then the determination can be carried out by a machine.

17 or 999999999999999 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 999999999999999 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.

Let us imagine the operations performed by the computer to be split up

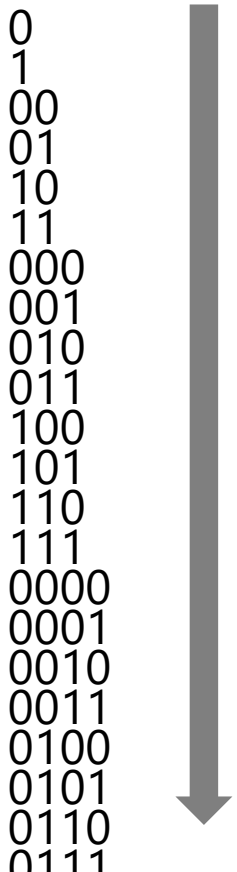
万能性と解決不能な問題

機械＝算譜（プログラム）＝算法（アルゴリズム）

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 $\delta(q, a) = (r, b, d)$ が有限個あるだけ

その文字列（算譜）を機械と呼ぶと思ってもよい



すべての機械は
このどこかに現れる

更に…

定理（万能チューリング機械）

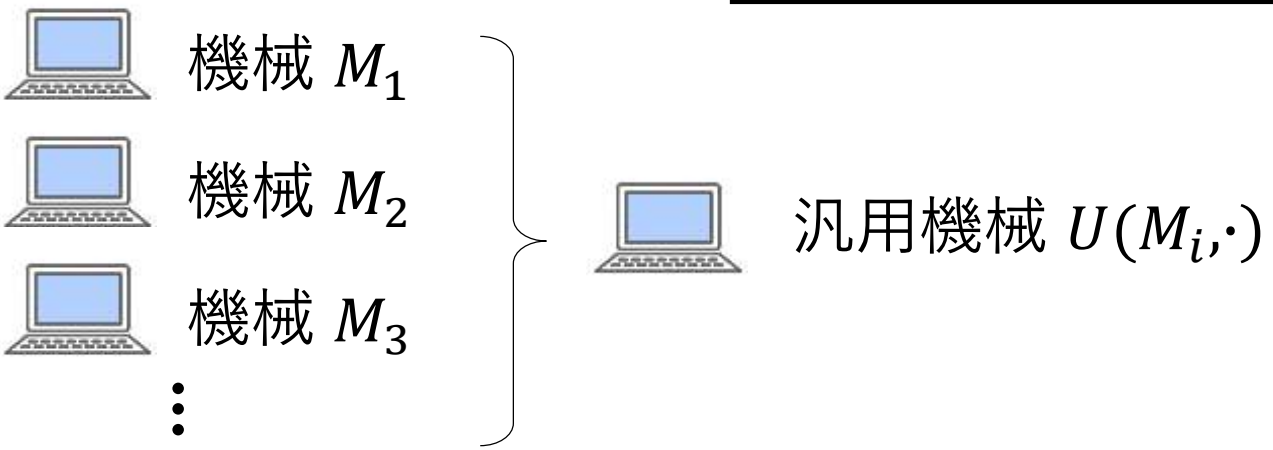
停止しないような
(M, x) については
U の動作もどうでもよい

次の問題を解く機械 U が存在

入力 機械 M と文字列 x の組 (M, x) であって
M に x を入力すると停止するもの

出力 M に x を入力したときの出力 $M(x)$ を出力

$$U(M, x) = M(x)$$



任意の機械 M の動きを $U(M, \cdot)$ として模倣（シミュレート）できる

汎用計算機（算譜（プログラム）内蔵式）を可能にする原理

定理（停止性判定問題は解けない）

次の問題を解く機械は**存在しない**

入力　機械 M と文字列 x の組 (M, x) ~~そあって~~
~~M に x を入力すると停止するもの~~

出力　M に x を入力したときの出力 $M(x)$ を出力　**但し**
M に x を入力して停止しない場合には「×」と出力

停止しない場合には
しないと明言する！

∵　もしこのような機械があれば　次を満す機械もあるはず

入力　機械 M と文字列 x の組 (M, x)

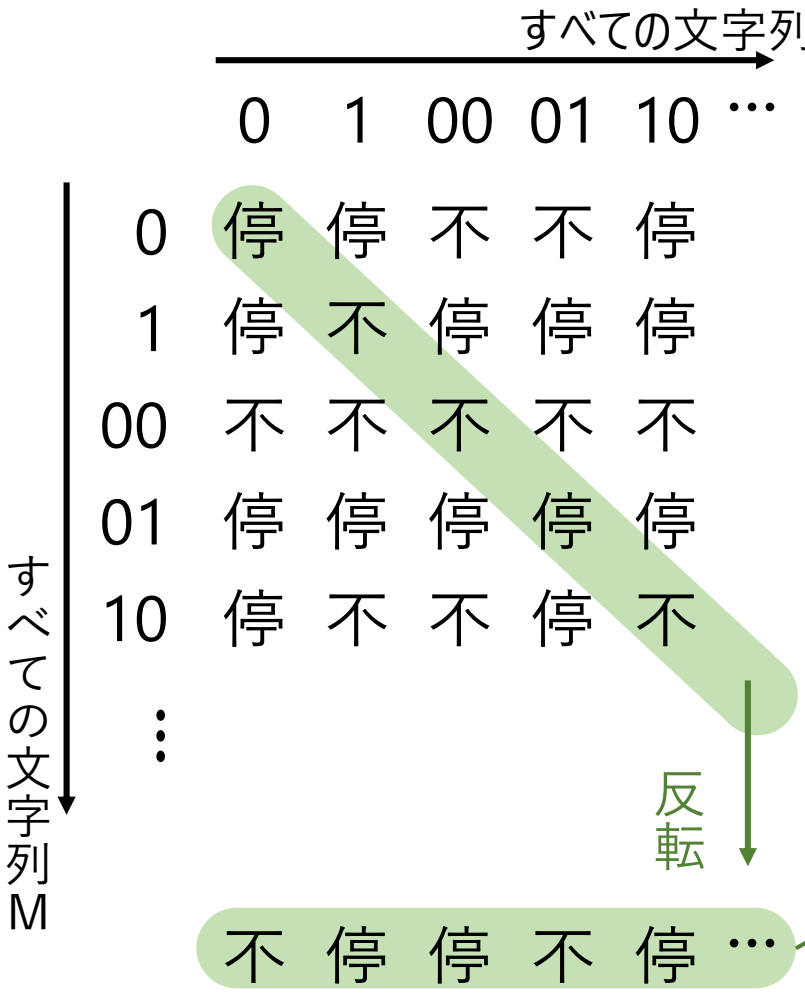
挙動　M に x を入力して停止するならば停止しない
M に x を入力して停止しないならば停止する

更に次を満す機械（これを**非停止性認識機械**と呼ぼう）もあるはず

入力　文字列 x

挙動　機械 x に x を入力して停止するならば停止しない
さもなくば停止する

しかしそのような機械は無い！
なぜなら…



対角線論法

（表紙でやったやつ）

「機械 M は入力 x で停止するか」を
第 M 行 x 列に記した表

非停止性の認識とは
この挙動をすることであった

しかしこれは表のどの行とも異なる
つまり**どの機械の挙動とも異なる**

非停止性の認識

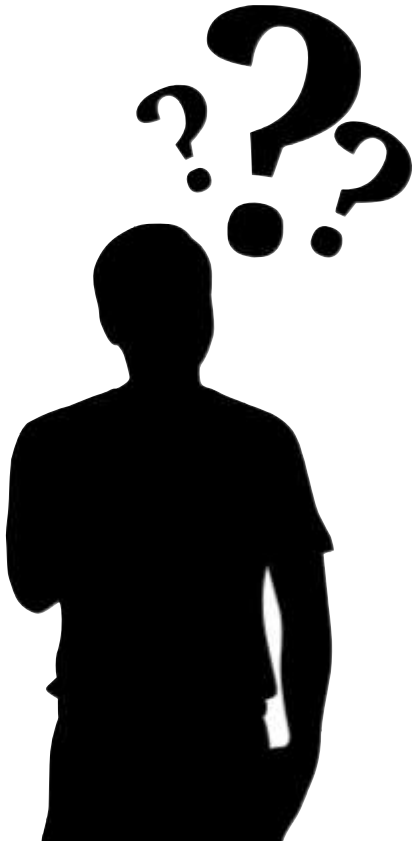
入力 文字列 x

挙動 機械 x に x を入力して停止するならば停止しない
さもなくば停止する

非停止性を認識する機械は無い

休憩

質問あればどうぞ



再

揃え不可能なときの動作が
「イイエと出力する」ではなく
「停止しない」で良ければ
この問題は解ける

ポストの文字列揃え問題

上下に文字列の書かれた札が何種類か与えられる
これらを横に並べて
上下の文字列を一致させることができるか？

入力				出力
b	a	ca	abc	ハイ
ca	ab	a	c	
acc	ca	abc		イイエ
cb	a	ab		

a	b	ca	a	abc
ab	ca	a	ab	c

とできるので

問題の難しさの比較（**帰着**）

ポストの文字列揃え問題

上下に文字列の書かれた札が何種類か与えられる
これらを横に並べて
上下の文字列を一致させることができるか？

ポストの文字列揃え問題（最初の札の指定つき）

上下に文字列の書かれた札が何種類か与えられ
その一つが指定されている

これらを横に並べて（**但し最初に使うのは指定された札とする**）
上下の文字列を一致させることができるか？

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

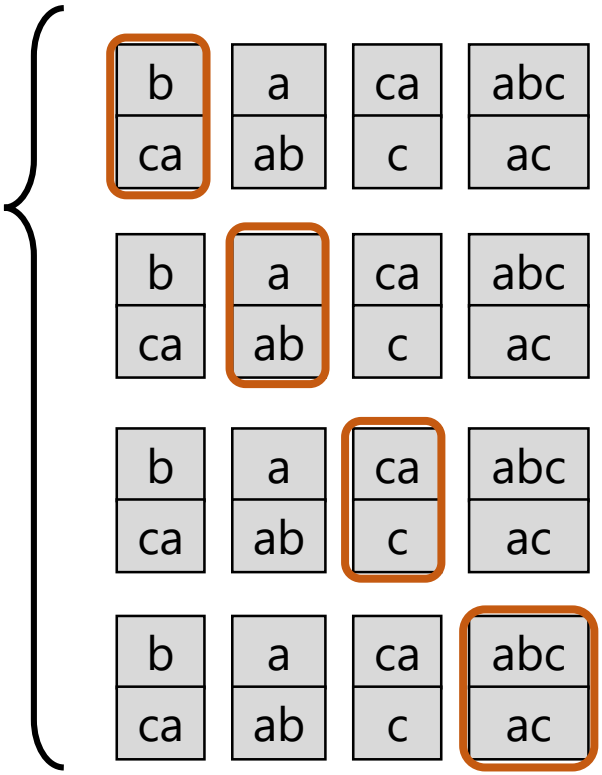
ポストの文字列揃え問題

帰着

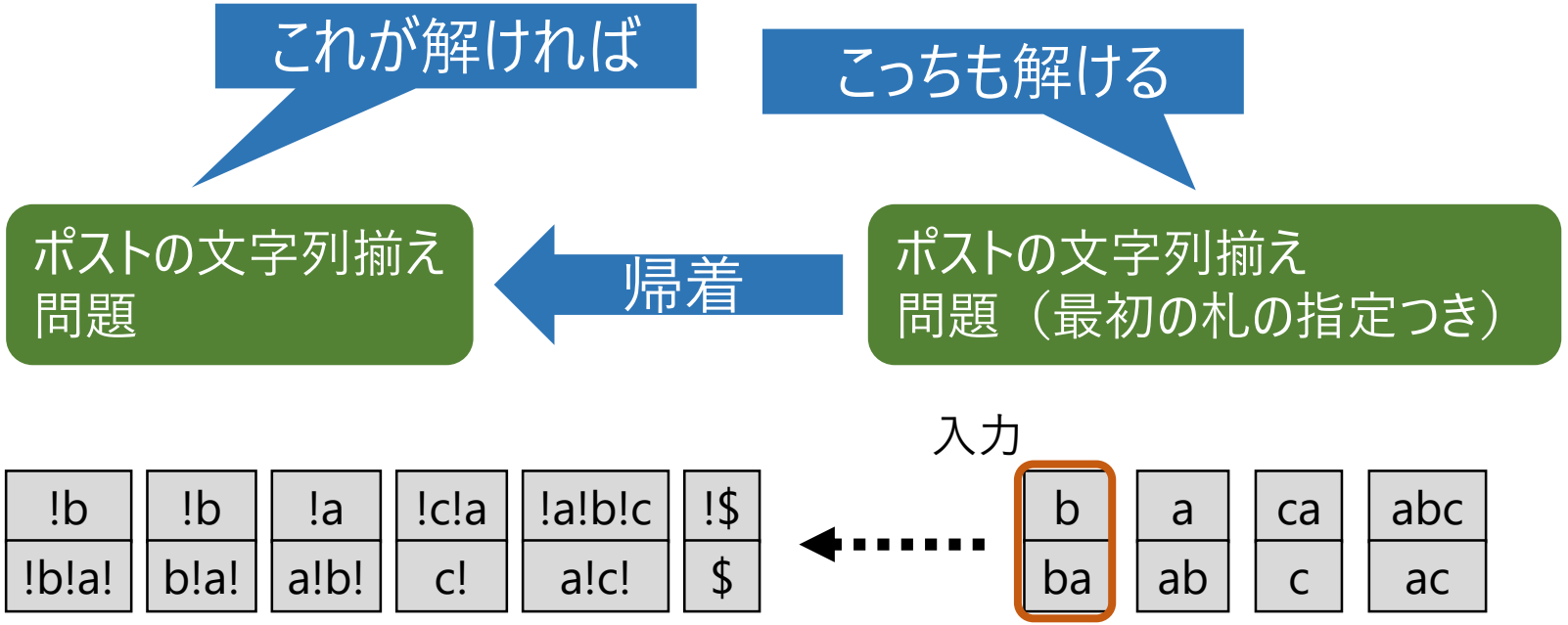
ポストの文字列揃え問題（最初の札の指定つき）

入力

b	a	ca	abc
ca	ab	c	ac



問題の難しさの比較（**帰着**）

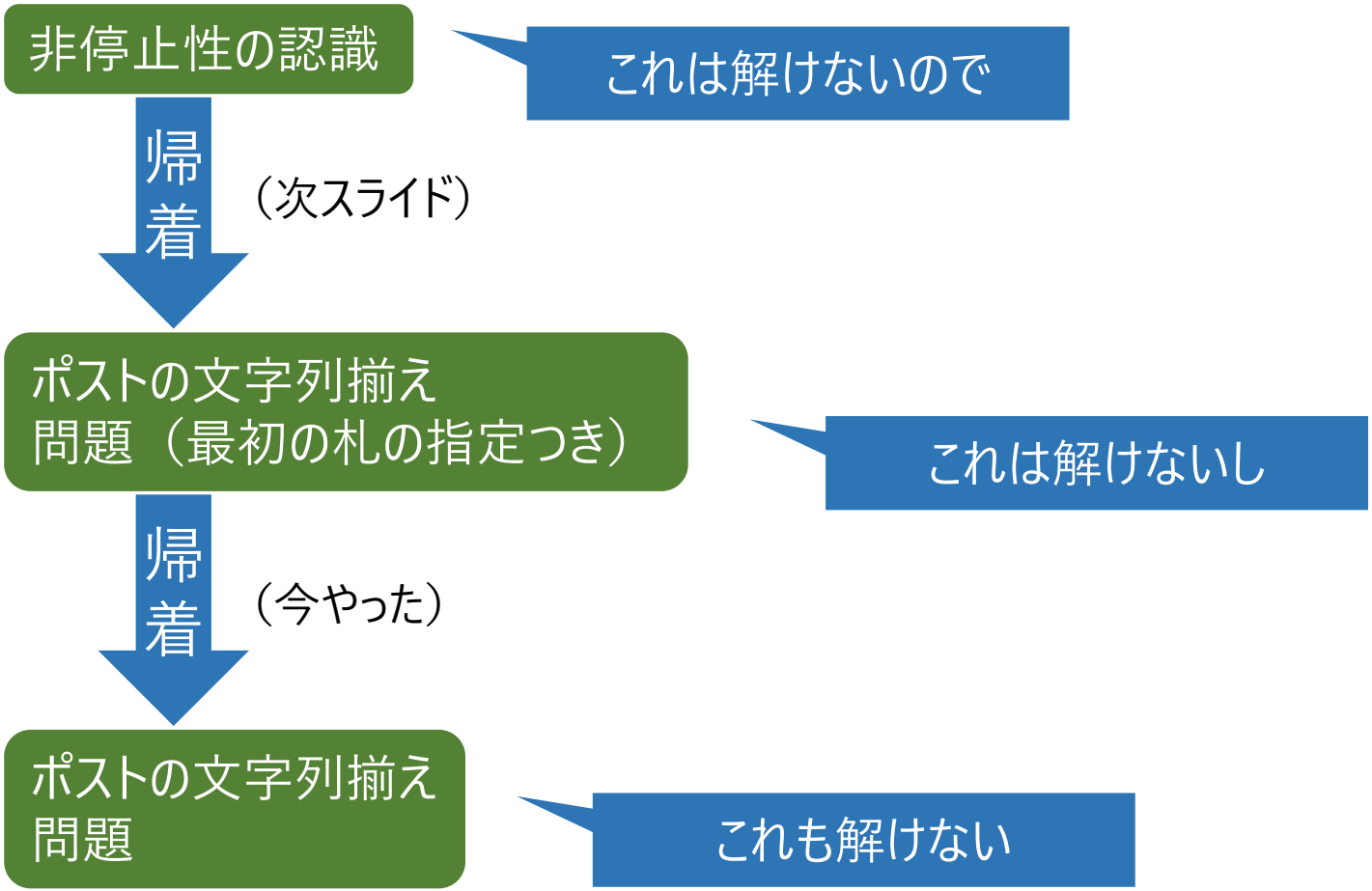


二つの問題は

（チューリング機械で解けるかどうかに関しては）

「同じ難しさ」であることが判った！

じつは……


非停止性の認識
これは解けないので
帰着
（次スライド）
ポストの文字列揃え
問題（最初の札の指定つき）
これは解けないし
帰着
（今やった）
ポストの文字列揃え
問題
これも解けない

非停止性の認識

帰着

非停止性の認識

（但し機械は停止するときは文字をすべて消してからテープ左端で状態 $q_{終}$ になるものとする）

帰着

ポストの文字列揃え問題（最初の札の指定つき）

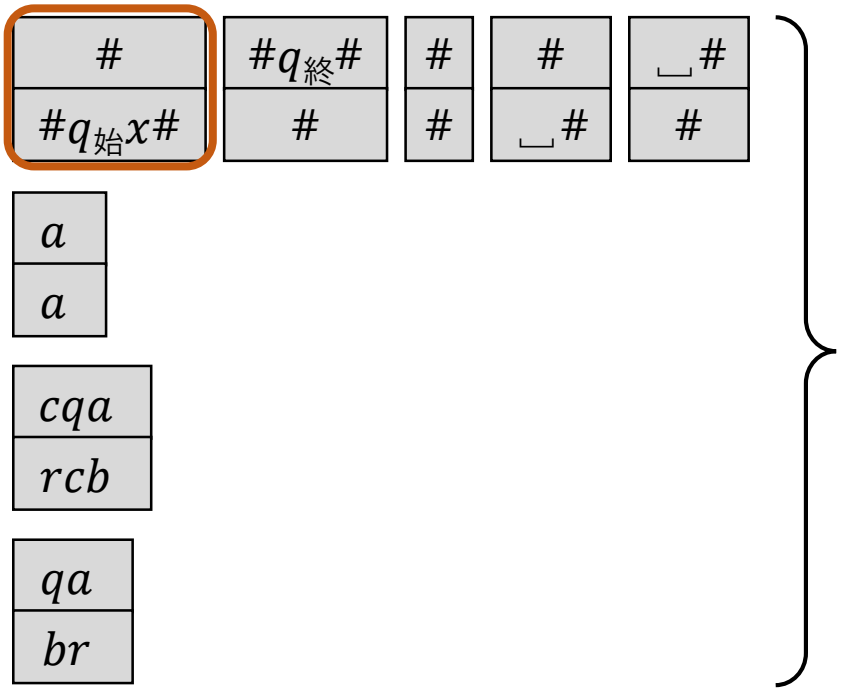
このように変換すれば

機械 M が入力 x で停止する ⟺ 上下揃えが完成できる

が成立つ

詳細略

(M, x)



M が使う各文字 a について

M の規則のうち $\delta(q, a) = (r, b, ㊧)$ という形のものそれぞれについて

M の規則のうち $\delta(q, a) = (r, b, ㊨)$ という形のものそれぞれについて

ヒルベルトの決定問題

一階述語論理の完全性

一階述語論理で書かれた主張（文）が与えられたとき
それが恒真（＝証明可能）か否かを判定する
機械的な手順はあるか？（1928）

Turing, A.M. (1936). "On Computable Numbers, with an Application to the Entscheidungsproblem". *Proceedings of the London Mathematical Society* 2, **42**: 230–265.

$\forall x.\forall z.\exists y.+(x,y)=z$

（整数と足し算の話なら正しいが…）

の機械

→「恒真でない」

$\forall x.\exists y.x =\bigstar (y) \wedge \forall y.\exists z.y =\blacktriangle (z)$
$\rightarrow \forall x.\exists z.x =\bigstar \left(\blacktriangle (z)\right)$

（どう解釈しても正しい）

の機械

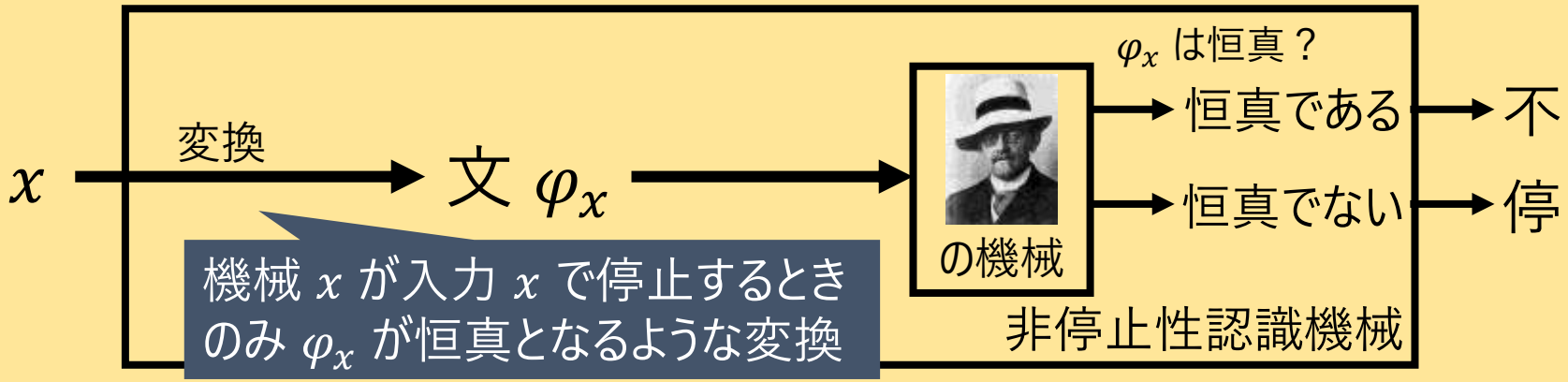
→「恒真である」

ヒルベルトの決定問題

一階述語論理で書かれた主張（文）が与えられたとき
それが恒真（＝証明可能）か否かを判定する
機械的な手順はあるか？（1928）

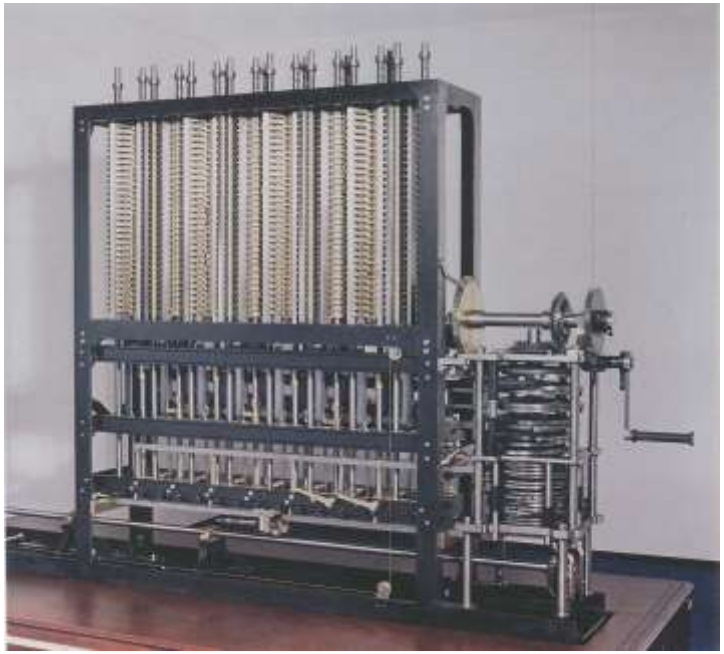
Turing, A.M. (1936). "On Computable Numbers, with an Application to the Entscheidungsproblem". *Proceedings of the London Mathematical Society* 2, **42**: 230–265.

もしあるとすると非停止性認識ができてしまうので、無い



計算量

As soon as an Analytic Engine exists, it will necessarily guide the future course of the science. Whenever any result is sought by its aid, the question will arise—By what course of calculation can these results be arrived at by the machine in the shortest time? — *Charles Babbage (1864)*

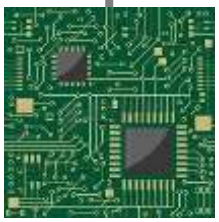


解析機関（Analytic Engine）

時間制限つき機械

$x =$ 0 0 0 1 1 0 1 1 0 1 1

テープ
（右側に無限）



機械 M

入力の長さ $|x|$ に応じた
制限時間 $t(|x|)$

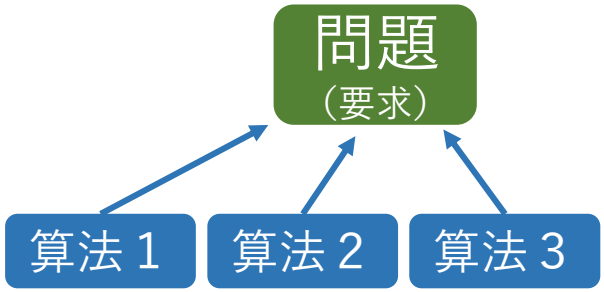
$t(n) = p(n)$（pは多項式）
であるような機械で
解ける問題の全体 **P**

$t(n) = 2^{p(n)}$（pは多項式）
であるような機械で
解ける問題の全体 **EXP**

- 何らかの文字列 $M(x)$ を書いて停止
- ~~停止しない~~

必ず $t(|x|)$ 回以内で停止

同じ問題を解くにも複数の方法があるが…



計算量の考え方（原則）

① チューリング機械の遷移の回数で測る
「現実の計算機上での基本操作の回数」にそこそこ近い

② 入力文字列の長さに関する函数として表す
（その長さの入力のうち最悪のものにかかる時間を考える）

③ その函数の増大の速さに着目
特に多項式以内か否かが重要

何故「多項式以内か否か」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				
$n!$	4秒	8百京年						

多項式時間

指数時間

1秒に1000000回の処理ができるとしたときにかかる時間

■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 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 の多項式時間で解く方法が見つかった

AKS素数判定法（2002）

問題 与えられたグラフ（頂点 n 個）がハミルトン閉路をもつか判定

頂点の並べ方（$n!$ 個ある）を全部調べれば判る

n の多項式時間で解く方法があるかどうかは判っていない

例　先程の問題と算法

問題　与えられた二つの正整数の最大公約数を求める

算法　エウクレイデスの互除法

```
procedure euclid(a, b) {
  r := (a を b で割った余);
  if (r == 0)
    return b;
  else
    return euclid(b, r);
  fi
}
```

動作例

$euclid(275, 500)$

$275 \div 500 = 0$　余275

$euclid(500, 275)$

$500 \div 275 = 1$　余225

$euclid(275, 225)$

$275 \div 225 = 1$　余50

$euclid(225, 50)$

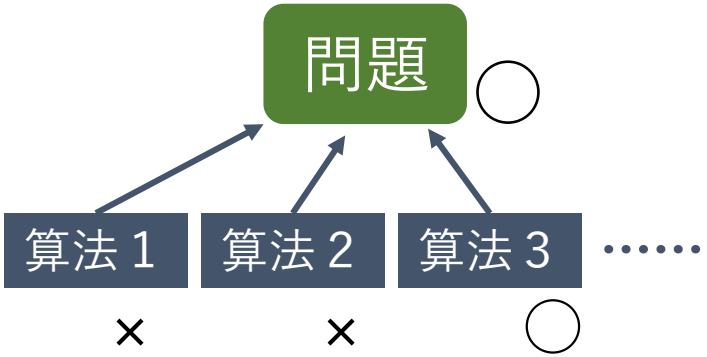
$225 \div 50 = 4$　余25

$euclid(50, 25)$

$50 \div 25 = 2$　余なし

25

これは多項式時間算法でしょうか？



問題□□は容易に解けるか？

＝

効率の良い算法が**存在する**か？

問題□□を（時間△△以内で）解くことは

如何なる算法を以てしても不可能　と証明したい

示すのが難しい

可能とも不可能とも示せない重要な問題は多い

それでも「時間制限つき帰着」を用いて問題の間の困難さを比べたり

それにより問題が「おそらく困難」と示したりできることはある

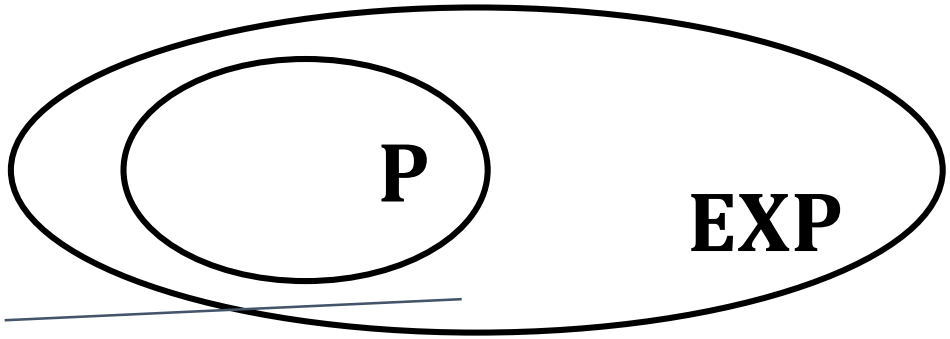
多項式時間で解けない問題

先程は（どんなに時間があっても）解けない問題を作ったが
少し構成法を変えることで

EXP には属するが **P** に属しない問題

が作れる

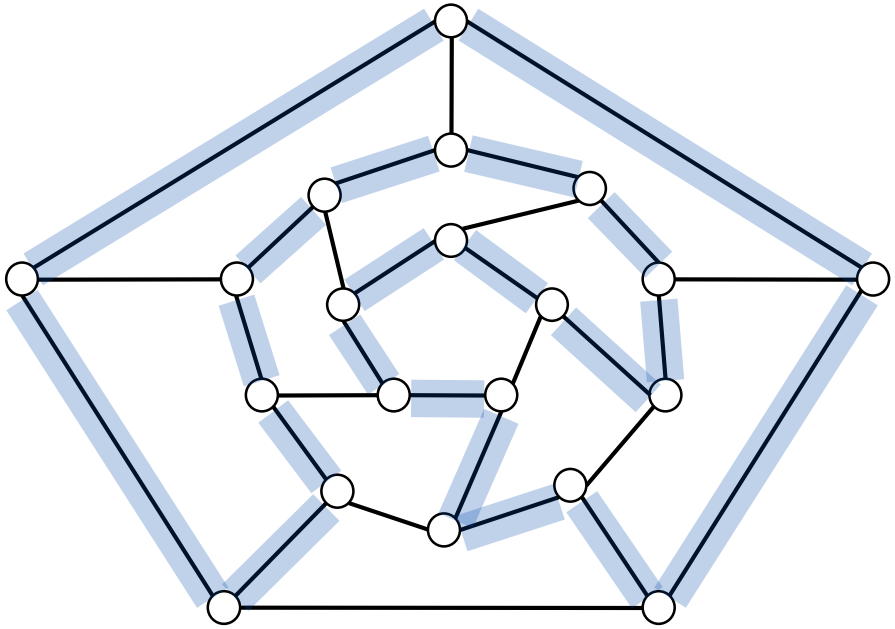
※ 判定問題に限定しても



同様に「$O(n^3)$ 時間で解けるが $O(n^2)$ 時間では解けない問題」の存在なども示される

このように「多くの時間を使うと
より多くの問題が解ける」という形の事実を**階層定理**という

一方で　**多項式時間算法が見つかってはいないのだが**
多項式時間算法が存在しないと証明されたわけでもない問題
もかなり多い



そのうち特に重要なものとして
NP 問題と総称されるものがある

先程の「ハミルトン路
存在判定問題」はその例

指数個の候補から何かを探す問題のうち
「候補を見せられればその適否が多項式時間で確認できる」もの
（正確な定義は省略）　（$\mathbf{NP} \subseteq \mathbf{EXP}$ であることはすぐ判る）

$\mathbf{P} \subseteq \mathbf{NP}$　（$\mathbf{P} = \mathbf{NP}$ であるか否かは未解決　百万ドル）

まとめ

このスライドは後で
講義ページにも載ります

- チューリング機械は何かが計算**できない**と示すために定義された

  「如何なる方法でも解けない」と示すには
  まず計算法とは何かを明らかにする必要があるため

- 非停止性の認識が不可能であることを　**対角線論法**により示した
- 他の問題も「もしそれが解けたら非停止性認識ができてしまうから」
  という理由により　解くことが不可能とわかることがある（**帰着**）
- 同様に多項式時間では解けない（が指数時間では解ける）問題が存在
- 「多項式時間では解けない」と証明されてはいなくても
  「問題○○が解ければ△△が解ける」という形で
  相対的な比較はできる場合がある（**時間制限つき帰着**）

レポート課題例　提出方法等については今井先生の指示に従って下さい

◆ 階層定理を示すには、講義で扱った対角線論法をどう修正すればよいか。
- 講義スライドでの停止性の認識不可能性の議論をもとに、それのどこをどう修正するか説明すること。

◆ 何らかの問題（例えばポストの文字列揃え問題）が（チューリング機械で）「解けない」或いは「解くことは○○以上に困難である」ことを帰着によって示す定理を一つ選び、如何なる議論によってそう示されるか概要を述べよ。
- 問題の内容（入力と出力）を明確に記すこと。
- どのように帰着するかは、概要のみでよいので、わかりやすさを優先してポイントを押さえた説明をせよ。

テーマはどれか一つでよい。自分で考えてもよいし、文献やウェブページなどを参考にしてもよいが、自分の理解に基づいて説明し、**参考とした文献等については適切な形で明記**せよ。
例えば以下の文献などが参考になるかもしれない。
- M・シプサ『計算理論の基礎』原著第二版（共立出版、平成20年）の第二巻
- 渡辺治『今度こそわかるP≠NP予想』（講談社、平成26年）

興味を持った人は

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）