大規模知識処理特論

第２回

脊戸　和寿

今回の講義内容

計算量理論の基礎について学ぶ

- Turing 機械
- 時間計算量と空間計算量
- クラス P とクラス NP

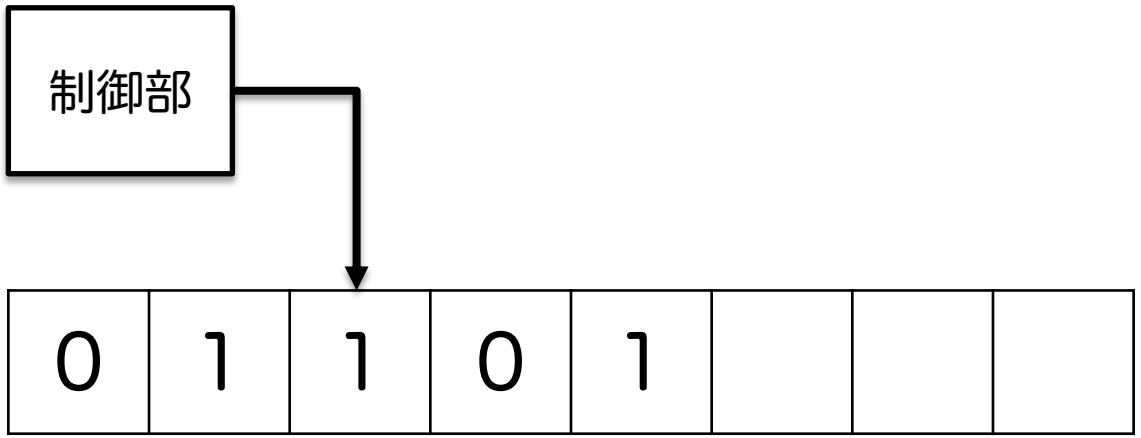
様々な分野で NP 困難やら NP 完全という用語が出てくるので、最低限のことは知っておいた方がよい。

Turing 機械

Turing 機械

計算機の数学的モデル

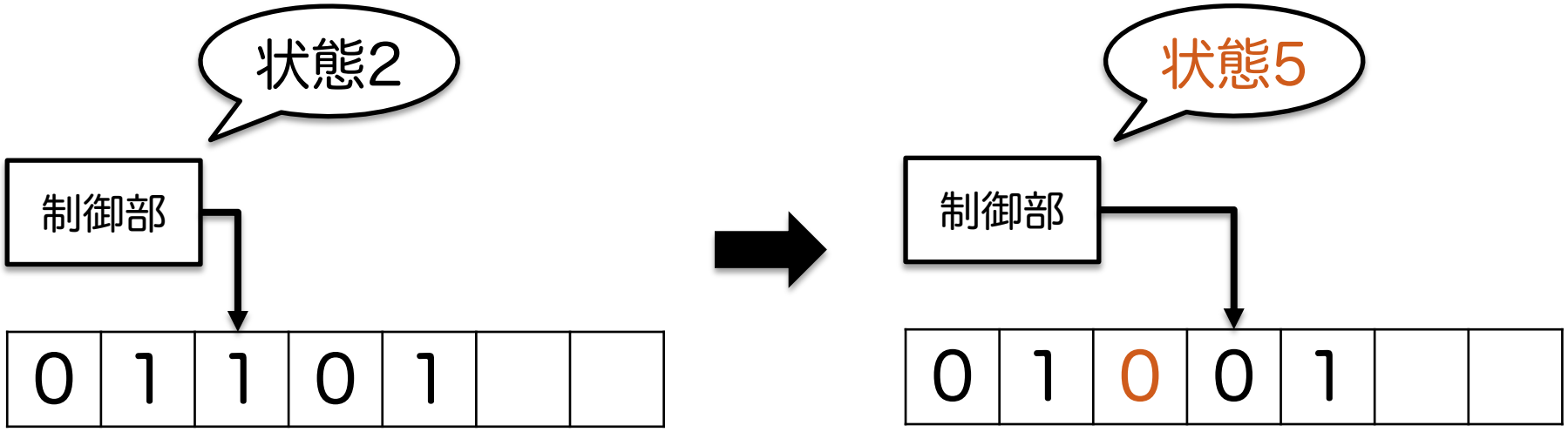
- 制御部とテープを持つ
- テープは無限の長さを持つ



Turing 機械

Turing 機械による計算

- テープの文字列へのアクセスや書き換えを行い、制御部内で状態を遷移させながら計算を行う。



Turing 機械による計算の例

$L = \{ w\#w \mid w \in \{0,1\}^* \}$ という言語（文字列の集合）を考える。

テープ上の文字列 s が L に属するなら受理、それ以外は非受理とする。

A) 1010#1010 → 受理

B) 01010#10101 → 非受理

C) 0101#10101 → 非受理（#より後ろの文字列の方が長い）

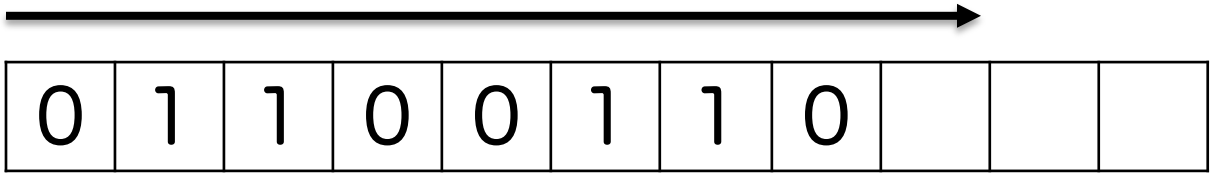
D) 11111#1111 → 非受理（#より前の文字列の方が長い ）

E) 00000000 → 非受理（#がない ）

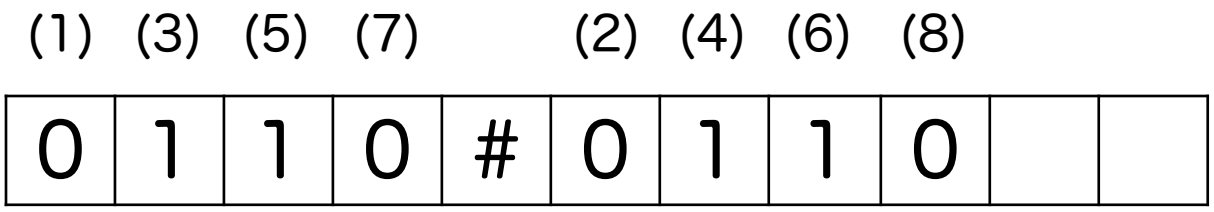
Turing 機械による計算の例

文字列 s が L に属するかを Turing 機械を使って判定する。

- # があるか走査して、なければ非受理
- #があれば先頭に戻って、 # の前と後ろの文字で対応する部分を確認していく。



がないので
非受理

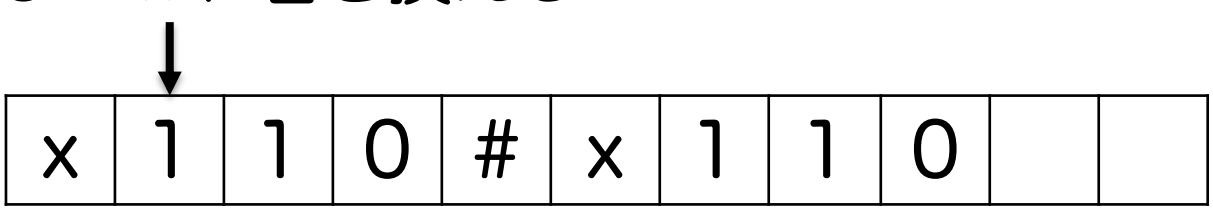


の前の文字列と
後ろの文字列を
交互に確認

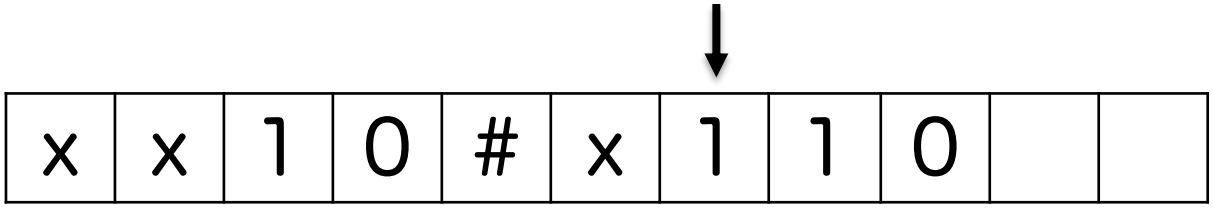
Turing 機械による計算の例

- # より左側では、文字を確認、記憶し、x に書き換える
- 右側では、記憶していた文字と一致しなければ、非受理
- 一致していれば、x に書き換えて、# より左側に戻る

記憶して x に書き換える



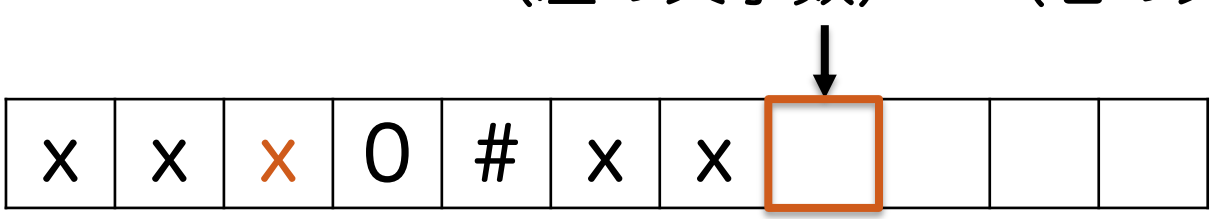
記憶した文字と一致しているので、
x に書き換えて # の左側へ



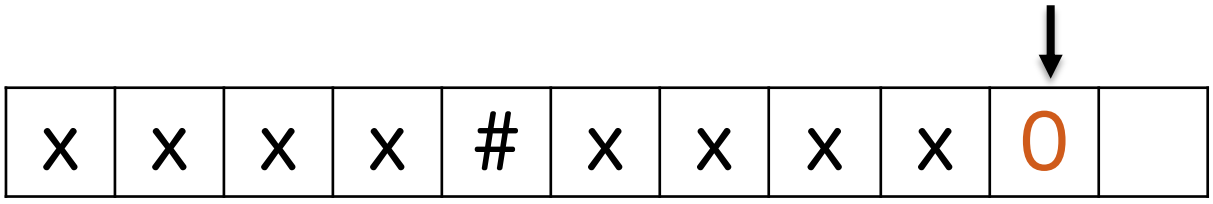
Turing 機械による計算の例

- 右側を見るときに対応する文字がなければ非受理
- 左側を全て見終わった後に、右側を全て走査する
  - ✓ 全部が x であれば受理、そうでなければ非受理

右側に 3 文字目がないので非受理
（左の文字数）＞（右の文字数）

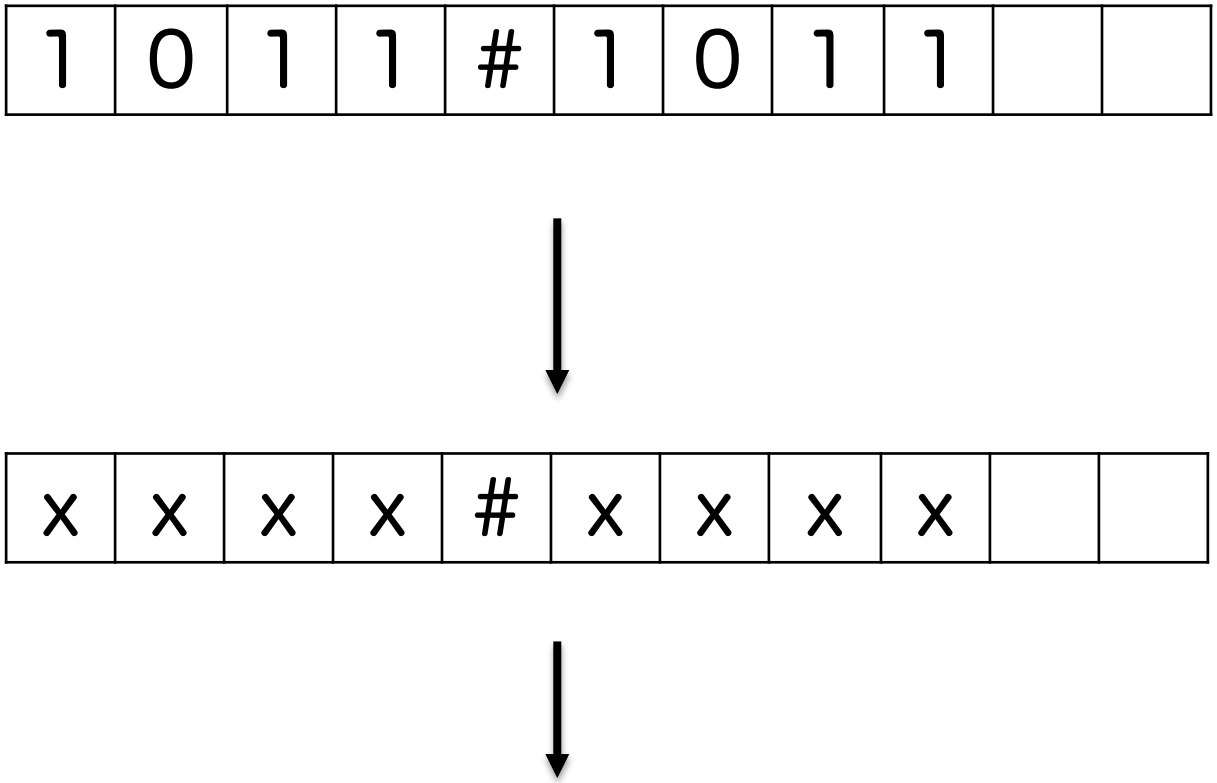


右側に x でない文字があるので非受理
（左の文字数）＜（右の文字数）



Turing 機械による計算の例

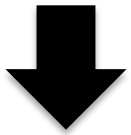
- L に属する文字列であれば、 # を挟んで同じ数の x が並ぶ



Turing 機械による計算

Turing 機械はオートマトン等の拡張

- ある言語を判定できる/できないを考えることが基本
- 判定できるのであれば、その言語を判定するために必要な計算ステップ数やテープの使用量を考える。



時間計算量や空間計算量

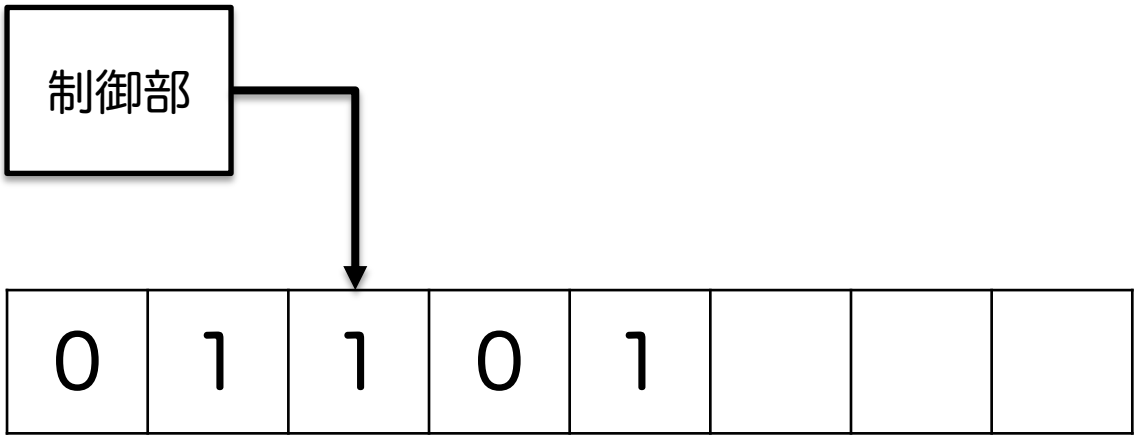
時間計算量と空間計算量

計算ステップ数

Turing 機械による計算におけるステップ数は？

- テープを 1 セル移動するために 1 ステップ
- テープの文字にアクセスするために 1 ステップ
- テープの文字を記憶するために 1 ステップ

……と、1 つ 1 つ考えていく必要がある



時間計算量

ある言語 L を Turing 機械で認識するためにかかるステップ数

- 入力長の関数として表す
- 最悪時間計算量と平均時間計算量がある
  - ✓ 最悪：受理/非受理を決定するために Turing 機械が要する最大ステップ数
  - ✓ 平均：すべての入力について，受理/非受理を決定するために Turing 機械が要するステップ数の平均をとったもの

ここでは、「時間計算量」を最悪時間計算量を表すことにする

ランダウ記法（オーダー記法）

１ つ １ つの操作を厳密なステップ数で議論をするのは難しい

→ 少しざっくりとした議論に落とす

例えば、

- １ bit 同士の加算は定数ステップで可能
- 入力長 n の列を全て確認するには最低 n ステップ必要

など

これらを $O(1)$，$\Omega(n)$ のように表す

→ このような記法をランダウ記法、オーダー記法という

ランダウ記法（オーダー記法）

O(・)記号の定義

ある自然数 n_0 と、ある正の定数 c が存在し、

自然数 $n \geq n_0$ に対し、

$$f(n) \leq cg(n)$$

を満たす時、$f(n) = O\big(g(n)\big)$、または $f(n) \in O\big(g(n)\big)$ と書く

【例】

$f(n) = 3n^2 + 2n + 2$、$g(n) = n^2$ のとき、$c = 4$、$n_0 = 3$ とすると、$f(n_0) = 35$、$cg(n_0) = 36$ より、$n \geq 3$ で、

$f(n) \leq cg(n)$ となるので，　$3n^2 + 2n + 2 = O(n^2)$ となる

ランダウ記法（オーダー記法）

Ω(・)記号の定義

ある自然数 n_0 と、ある正の定数 c が存在し、

自然数 $n \geq n_0$ に対し、

$$f(n) \geq cg(n)$$

を満たす時，$f(n) = \Omega\big(g(n)\big)$，または $f(n) \in \Omega\big(g(n)\big)$ と書く

【例】

$f(n) = n^2 + n - 2$、$g(n) = n^2$ のとき、$c = 1$、$n_0 = 2$ とすると、$f(n_0) = 4$、$cg(n_0) = 4$ より、$n \geq 2$で、$f(n) \geq cg(n)$ となるので、$n^2 + n - 2 = \Omega(n^2)$ となる

ランダウ記法（オーダー記法）

O(・), Ω(・)は直感的には,

- $f(n) = O\big(g(n)\big)$

  $f(n)$ の増加速度は早くとも $g(n)$ の増加速度くらい

- $f(n) = \Omega\big(g(n)\big)$

  $f(n)$ の増加速度は遅くとも $g(n)$ の増加速度くらい

また，$f(n) = O\big(g(n)\big)$ かつ $f(n) = \Omega\big(g(n)\big)$ のとき，$\Theta\big(g(n)\big)$ と書く

ランダウ記法（オーダー記法）

O も Ω も最も増加の速い項のみ（係数も除く）を残すことで求めることができる.

【例】

- $f(n) = 3n^4 + 2n^2 + 3 = O(n^4)$
- $f(n) = 0.2n\log_2 n + 4n - 2 = O(n\log_2 n)$
- $f(n) = 3 \cdot 2^{0.2n} + 2n^{\log_2 n} = 3 \cdot 2^{0.2n} + 2^{(\log_2 n)^2+1} = O(2^{0.2n})$

ランダウ記法（オーダー記法）

先ほどの例

- 1 bit 同士の加算は定数ステップで可能 $\rightarrow O(1)$ ステップ
- 入力長 n の列を全て確認するには n ステップ必要 $\rightarrow \Omega(n)$ ステップ

その他にも

- n bit 同士の加算 $\rightarrow O(n)$ ステップ
- n bit 同士の乗算 $\rightarrow O(n^2)$ ステップ

など

時間計算量

$L = \{ w\#w \mid w \in \{0,1\}\}$ を認識する Turing 機械の時間計算量

$\rightarrow O(|s|) + O(|s|) \times O(|s|) + O(|s|) = O(|s|^2)$、$s$：入力列

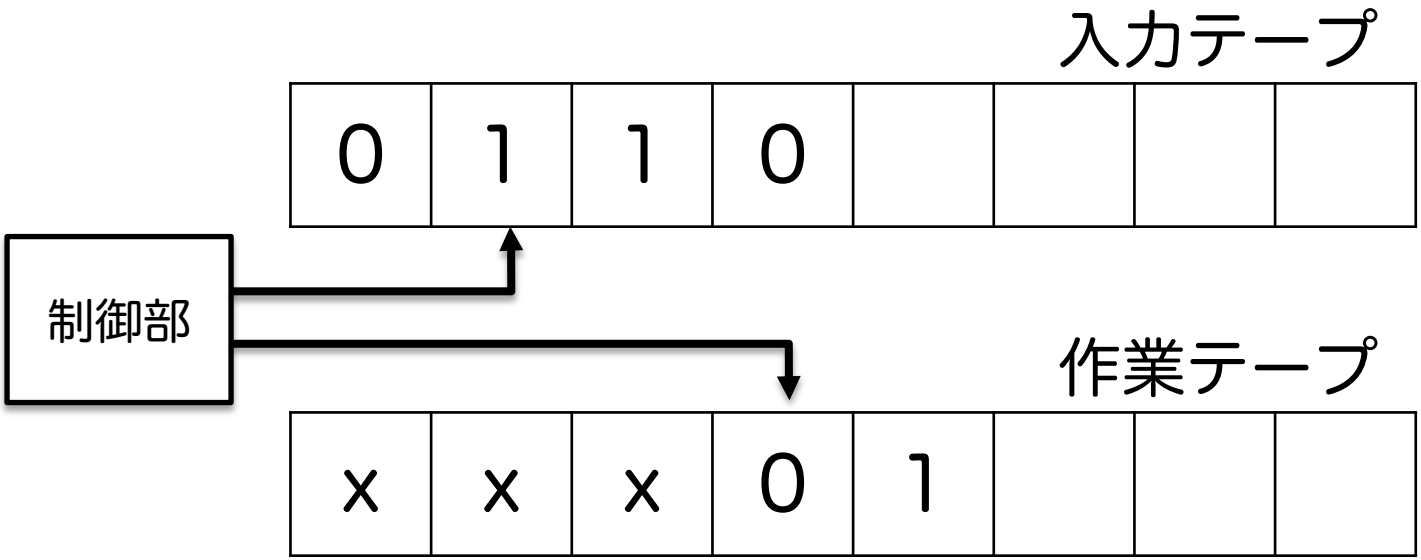
受理される入力例を考えると、入力長 $|s| = 2|w| + 1$

- # があるか走査して先頭に制御部が戻る $\rightarrow O(|s|)$ ステップ
- 先頭の文字、$|w| + 2$ 番目の文字，2 番目の文字，$|w| + 3$ 番目の文字と順々に確認していく $\rightarrow O(|s|^2)$ ステップ
  - ✓ 1 つのペアの確認 $\rightarrow O(|s|)$ ステップ
  - ✓ 全部で |w| ペア $\rightarrow O(|s|)$ 回の繰り返し
- 先頭から後ろまで x が続くか確認する $\rightarrow O(|s|)$ ステップ

空間計算量（領域計算量）

2 つのテープもつ Turing 機械を考える

- 入力テープ：入力列が書かれた読み取り専用のテープ
- 作業テープ：計算の作業に使える読み書き可能なテープ



空間計算量は、計算に使用する「作業テープ」の使用量を表す

練習問題1

次の関数 $f(n)$ をオーダー表記 O(・) であらわせ

- $f(n) = 3$
- $f(n) = 2\log_2 n + n + 1$
- $f(n) = 3\sqrt{n} + n^{0.3}\log_2 n$
- $f(n) = \log_2 n + n^{\epsilon}\ (\epsilon > 0)$
- $f(n) = 2^{0.1\sqrt{n}} + 2^n$
- $f(n) = 2^n + n!$

練習問題1解答

次の関数 $f(n)$ をオーダー表記 O(・) であらわせ

- $f(n) = 3 = O(1)$
- $f(n) = 2\log_2 n + n + 1 = O(n)$
- $f(n) = 3\sqrt{n} + n^{0.3}\log_2 n = O(\sqrt{n})$
- $f(n) = \log_2 n + n^{\epsilon}\ (\epsilon > 0) = O(n^{\epsilon})$
- $f(n) = 2^{0.1\sqrt{n}} + 2^n = O(2^n)$
- $f(n) = 2^n + n! = O(n!)$

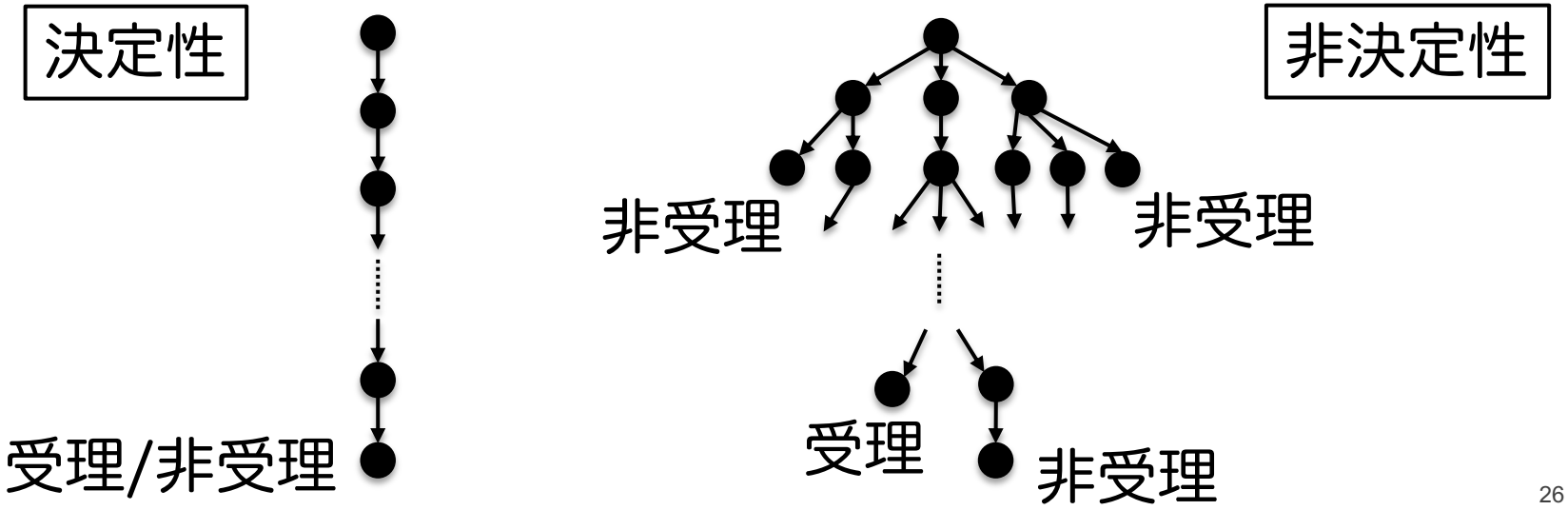
クラス P とクラス NP

決定性と非決定性

Turing 機械による計算に決定性と非決定性計算がある

入力された列に対し、

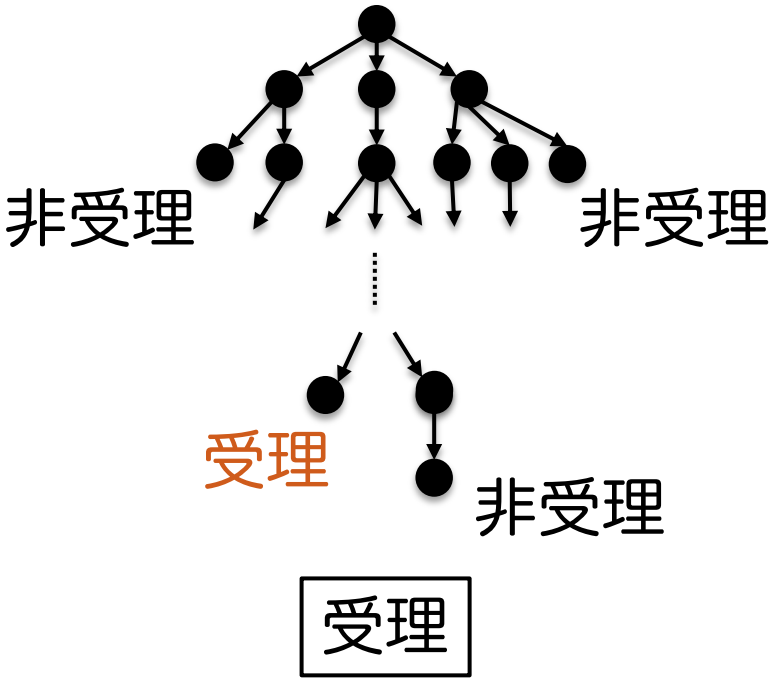
- 決定性：計算の最終状態が一意に決まる
- 非決定性：計算の最終状態が複数存在することがある

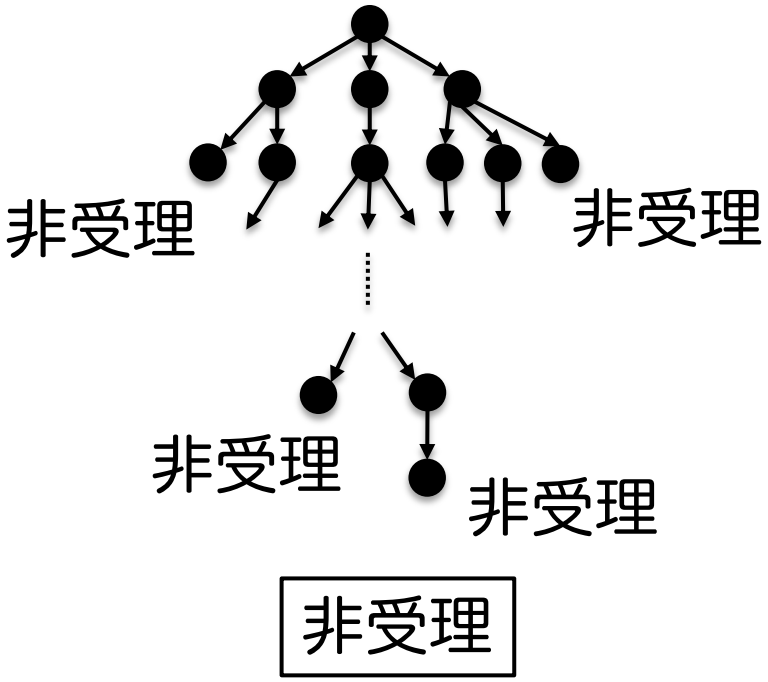


決定性と非決定性

列の受理 / 非受理は？

- 決定性：最終状態で受理/非受理が決定可能.
- 非決定性：ある計算パスで「受理状態」があれば受理.

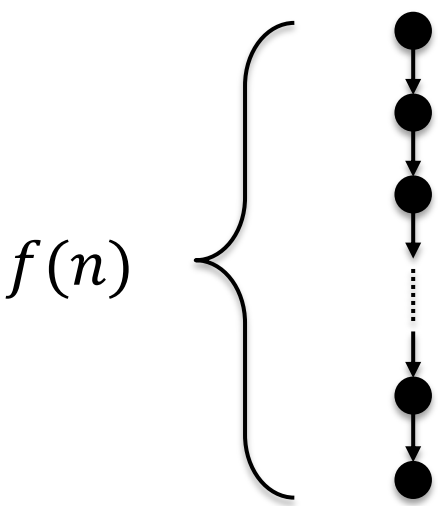


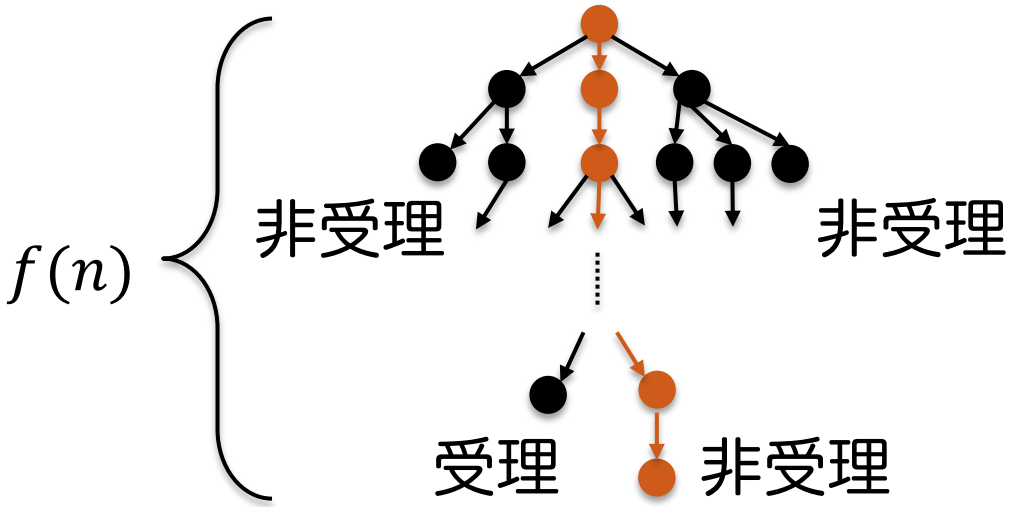


決定性と非決定性

計算時間：$f(n)$

- 決定性：計算パスの長さ
- 非決定性：最も長い計算パスの長さ





決定問題

決定問題：答が Yes / No（１/０）に限られた問題.

x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

決定問題

x	y	f
1	1	0
1	2	1
1	3	1
2	1	1
2	2	0
2	3	1
3	1	1
3	2	1
3	3	0

決定問題

x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	2
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	2
1	1	1	1

決定問題ではない

クラス P とクラス NP

クラス P（ Polynomial time ）

- 決定性 TM で多項式時間で解ける決定問題の集合

クラス NP（ Nondeterministic Polynomial time ）

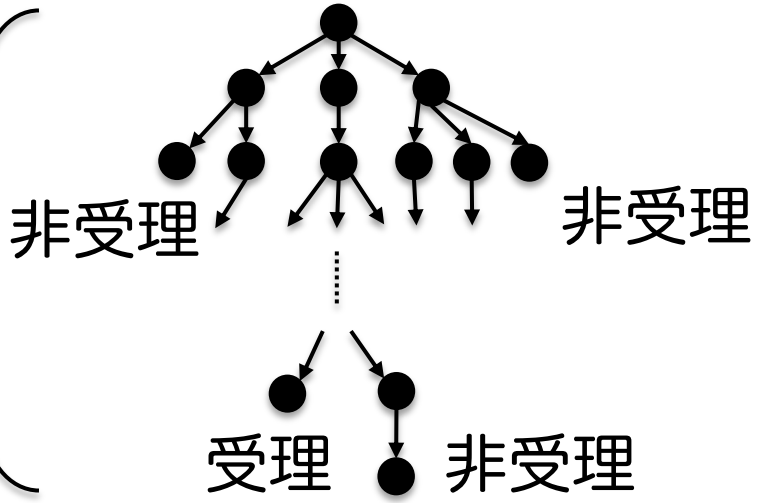
- 非決定性 TM で多項式時間で解ける決定問題の集合

$f(n) = O(n^k)$

kは定数

$f(n) = O(n^k)$

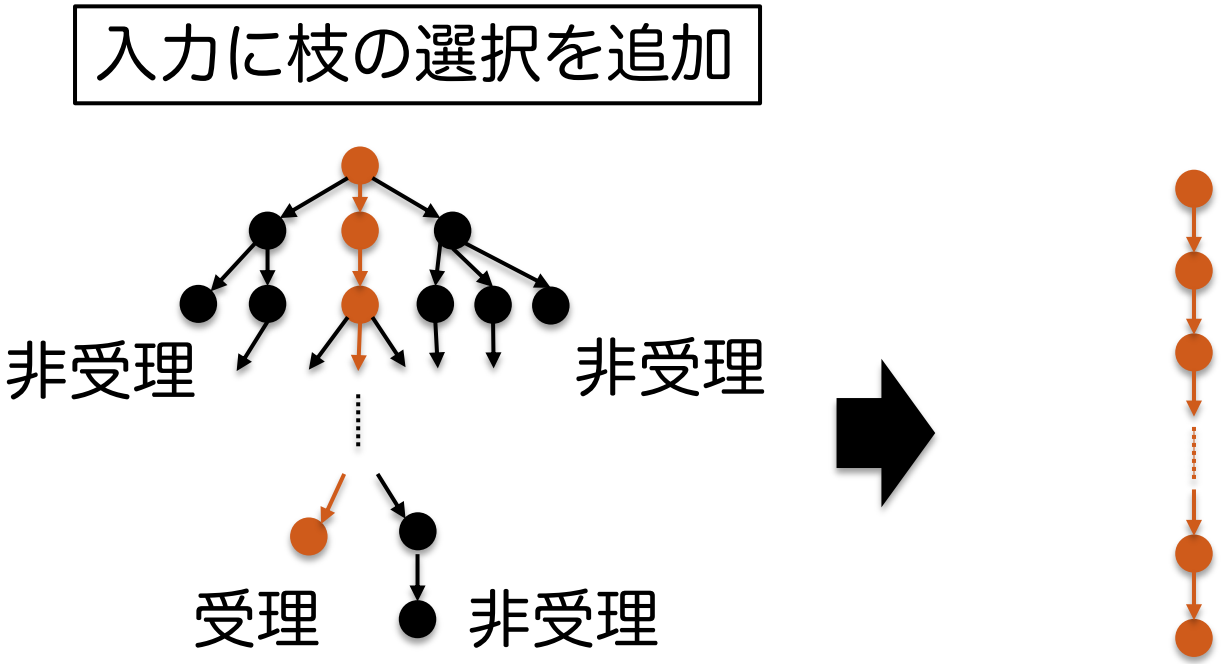
kは定数



クラス NP

クラス NP には，もう 1 つの定義がある．

- 非決定計算でどの枝を辿ればよいかという情報を与えると、決定計算に変わる



クラス NP

クラス NP には，もう 1 つの定義がある.

- Yes となる列 x と証拠（witness）w を入力として与えたとき、$f(|x| + |w|) = O((|x| + |w|)^k)$ 時間で Yes と判定できる決定問題の集合。ただし、$|w| = O(|x|^c)$ である。c, k は定数。

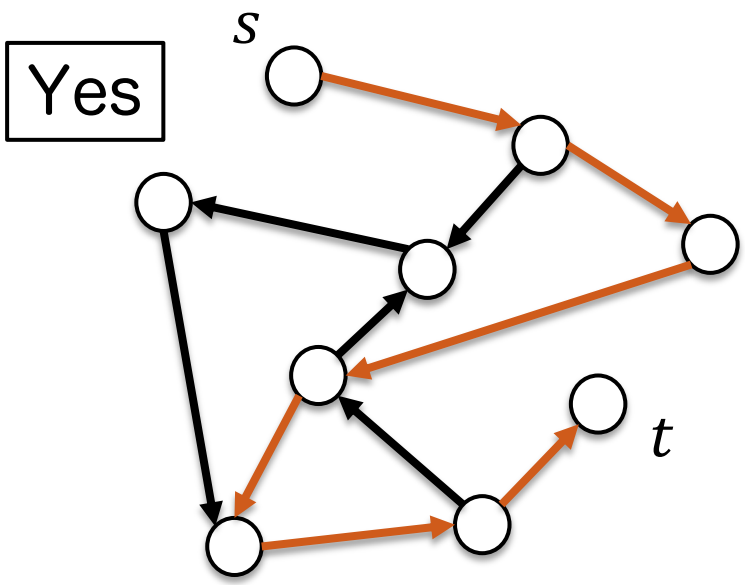
ざっくり言うと、問題とその解を与えたときに、その解が問題の解として正しいかを多項式時間で検証できるクラス。

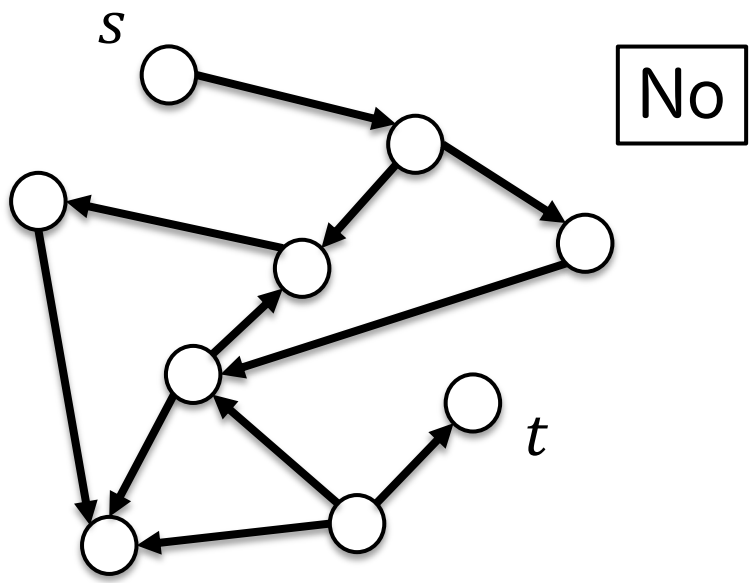
クラス P の例

<u>パス問題</u>

入力：有向グラフ G と頂点 s, t.

問：s から t に至るパスが存在するか？





深さ優先探索で $O(|G|)$ で解くことが可能.

クラス P の例

<u>素数判定問題</u>

入力：自然数 N

問：N は素数か？

試し割りで $O(\sqrt{N})$ で解くことが可能 → P に属する？

- N を 0/1 列で表すために、$[\log_2 N]$ bit 必要なので、入力長は $O(\log_2 n) \rightarrow O\left(\sqrt{N}\right) \notin O\left((\log_2 n)^k\right)$

$O\left((\log_2 n)^{\frac{15}{2}}\right)$ アルゴリズム [Agrawal et al. 2004]

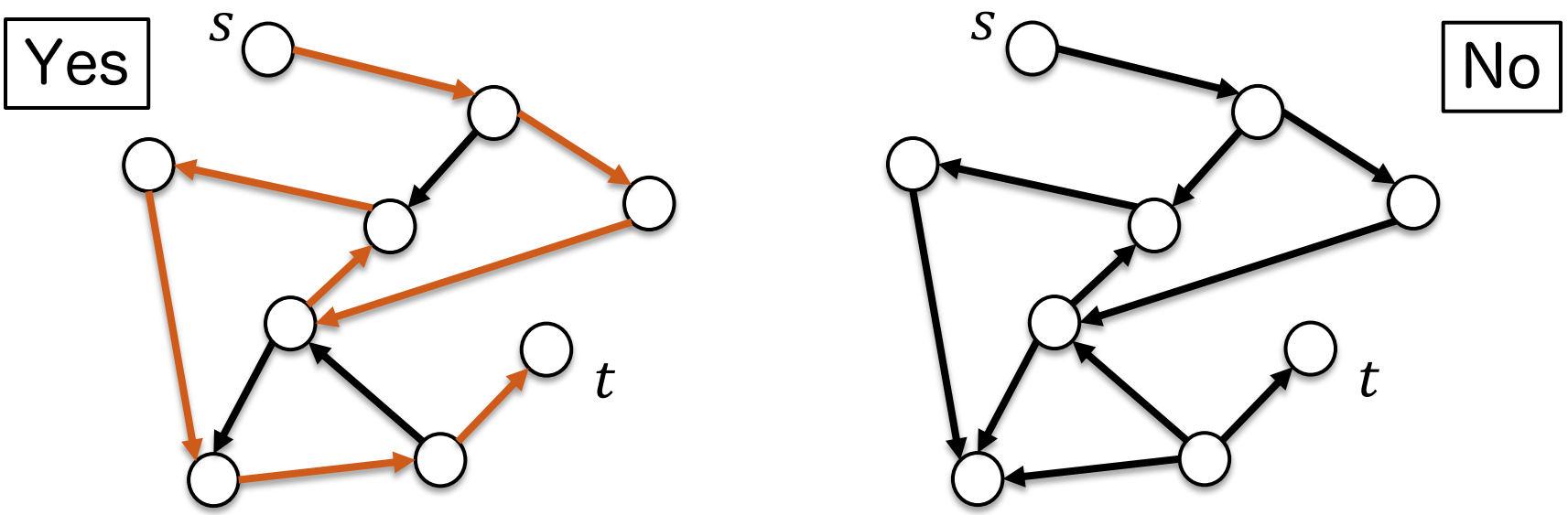
クラス NP の例

ハミルトンパス問題

入力：有向グラフ G と頂点 s, t.

問：s から t に至るハミルトンパスが存在するか？

- ハミルトンパス：全ての頂点をちょうど 1 度だけ通るパス



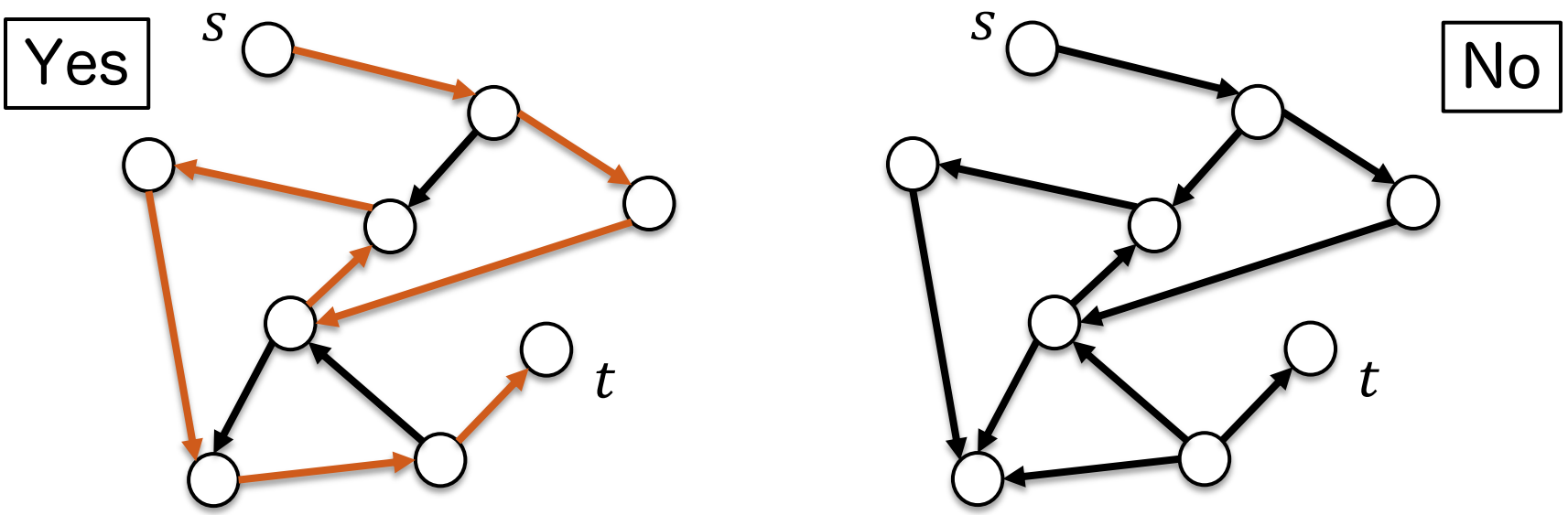
クラスNPの例

ハミルトンパス問題

入力：有向グラフ G と頂点 s, t.

問：s から t に至るハミルトンパスが存在するか？

- ハミルトンパス：全ての頂点をちょうど 1 度だけ通るパス



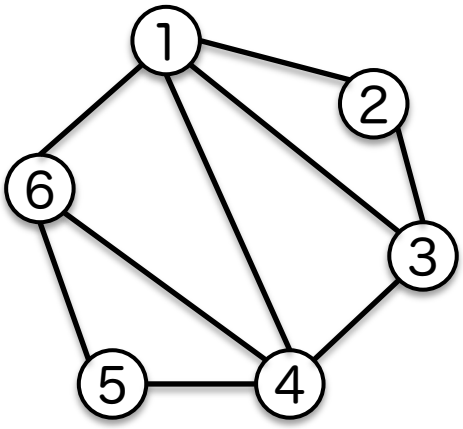
クラス NP の例

<u>グラフ同型性問題</u>

入力：無向グラフ G と H

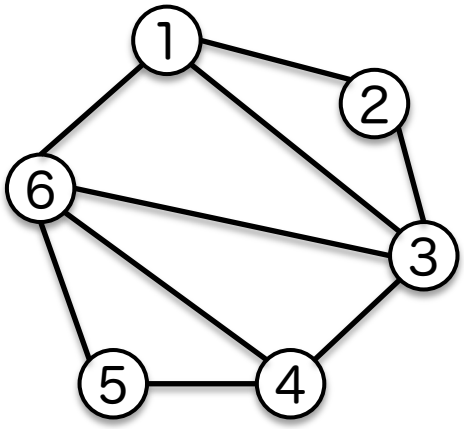
問：G と H は同型か？（H の頂点番号を入れ替えると、 G と同じ形のグラフになるか）

$1 \Leftrightarrow 6$
$2 \Leftrightarrow 5$
$3 \Leftrightarrow 4$

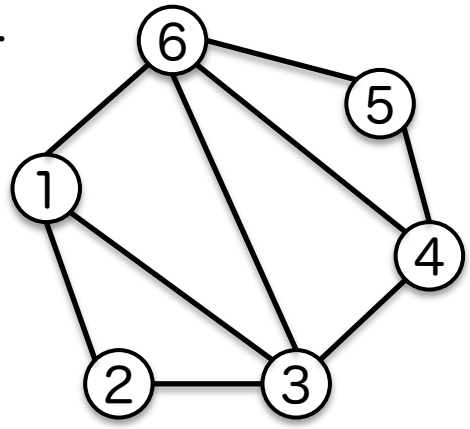


G

Yes



H



P vs. NP 問題

P vs. NP 問題：クラス P とクラス NP は等しい？

- クレイ数学研究所により、ミレニアム懸賞金問題に指定
- $P \subseteq NP$ は自明（非決定性計算は決定性計算を含むため）
- $NP \subseteq P$ か？

どちらを証明してもよい

- $P \neq NP$：NP に属する問題に対し、P に属さないことを示す
- $P = NP$：NP 完全問題の多項式時間アルゴリズムを設計

NP 完全問題については，次回の講義で扱う

まとめ

Turing機械とはコンピュータの数学的モデル

- 決定性と非決定性

時間計算量と空間計算量

- ランダウの記号

クラス P と NP、P vs. NP 問題

- 問題を実際に解くことと解を検証することの違い
- P と NP に属する問題の例