第4章　計算の複雑さ入門

4.1. 計算の複雑さの理論概観

「計算可能か？」➔「どの程度の計算コストで計算可能か？」
計算の複雑さの理論 (Computational Complexity Theory)

定義4.3: 自然数上の関数 f, g に対し,
$\exists c,d > 0, \forall n\ [f(n) \leqq c\, g(n) + d]$
となるとき, f は<u>オーダーgである</u>といい, $f = O(g)$ と記述する.

★定数c, dはnと無関係に定まることが必要.

定理4.1: 自然数上の任意の関数 f, g, h に対し次の関係が成立。
(1) $\forall n[f(n) \underset{\infty}{\leqq} g(n)] \rightarrow f = O(g)$
(2) $\exists c > 0, \forall n[f(n) \leqq cg(n)] \rightarrow f = O(g)$
(3) $[f = O(g)$ かつ $g = O(h)] \rightarrow f = O(h)$

Chap.4 Computational Complexity

4.1. Survey on Theory of Computational Complexity

"Computable?" ➔ "How much cost is required for computation?

Computational Complexity Theory

Definition 4.3: For functions f and g on natural numbers, if
$\exists c,d > 0, \forall n\ [f(n) \leqq c\, g(n) + d]$
then we say f is <u>in the order of g</u> and denote it by $f = O(g)$.

Remark: the constants c and d must be determined independently of n.

Theorem 4.1: The followings hold for any functions f, g and h on natural numbers:

1. $\forall n[f(n) \leqq g(n)] \rightarrow f = O(g)$
2. $\exists c > 0, \overset{\infty}{\forall} n[f(n) \leqq cg(n)] \rightarrow f = O(g)$
3. $[f = O(g)$ and $g = O(h)] \rightarrow f = O(h)$

4.2.3. 問題の時間計算量

定義4.4. Φ を計算問題とし，t を自然数上の関数とする.
いま Φ を計算するプログラム A と定数 $c, d > 0$が存在して，

$$\forall l\,[time\_A(l) \leqq ct(l) + d]$$

ならば，Φは$O(t)$時間計算可能，あるいはΦの時間計算量は$O(t)$であるという.

注意:ここでは計算問題として，集合の認識問題を想定している.

直観的には「問題Φは t 時間以下で計算可能」という意味。

(注1) A の時間計算量は t より低いかもしれない.
(注2) A よりも速くΦを計算するプログラムがあるかもしれない.

4.2.3. Time complexity of a problem

Def.4.4. Let Φ be a computing problem and t be a function over natural numbers. If we have a program A to compute Φ and some constants c and $d > 0$ such that

$$\forall l \, [time\_A(l) \leqq ct(l) + d]$$

then we say that Φ is computable in $O(t)$ time, or time complexity of Φ is $O(t)$.

Notice: We assume here that a computing problem is that of recognizing a set.

Intuitively

problem Φ is computable within time t

- time complexity of A may be less than t.
- there may be a faster program to compute Φ than A does.

例4.7. 素数判定問題の時間計算量

素数判定問題(PRIME)
入力：自然数 n（ただし，2進表記）
質問：n は素数か？
PRIME $\equiv \{ \lceil n \rceil : n$は素数$\}$

スターリングの公式：

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

```
prog Naive(input n);
begin
  for each i := 1 < i < n do
    if n mod i = 0 then reject end-if
  end-for;
  accept
end.
```

2 ～ n-1の数で割ってみる

$\log n \cdot \log i$ 時間

余談：
2002年に
$O(l^6)$
のアルゴリズム
が考案された!!

$$time\_\mathrm{Naive}(n) \le \sum_{1<i<n} (c\log n \log i + d)$$
$$= c\log n \log n! + dn = O(n(\log n)^2)$$

n の長さを l とすると，l はほぼ$\log n$だから，$time\_\mathrm{Naive}=O(l^2 2^l)$
故に，素数判定問題の時間計算量は（高々）$O(l^2 2^l)$

Ex.4.7. Time complexity of the problem determining primes

Prime-determining problem(PRIME)
Input : a natural number n (binary representation)
Question: Is n prime?
PRIME $\equiv \{ \lceil n \rceil : n \text{ is prime}\}$

Stirling's Formula :

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

try to divide by numbers between 2 – n-1

```
prog Naive(input n);
begin
  for each i:= 1 < i < n do
    if n mod i = 0 then reject end-if
  end-for;
  accept
end.
```

log n ▪ log i time

$O(l^6)$ time algorithm has been developed in 2002!!

$$time\_\text{Naive}(n) \le \sum_{1<i<n} (c \log n \log i + d)$$
$$= c \log n \log n! + dn = O(n(\log n)^2)$$

When the length of n is l, l is approximately log n. So, $time$_Naive $=O(l^2 2^l)$. Thus, time complexity of PRIME is $O(l^2 2^l)$.

定義4.5.

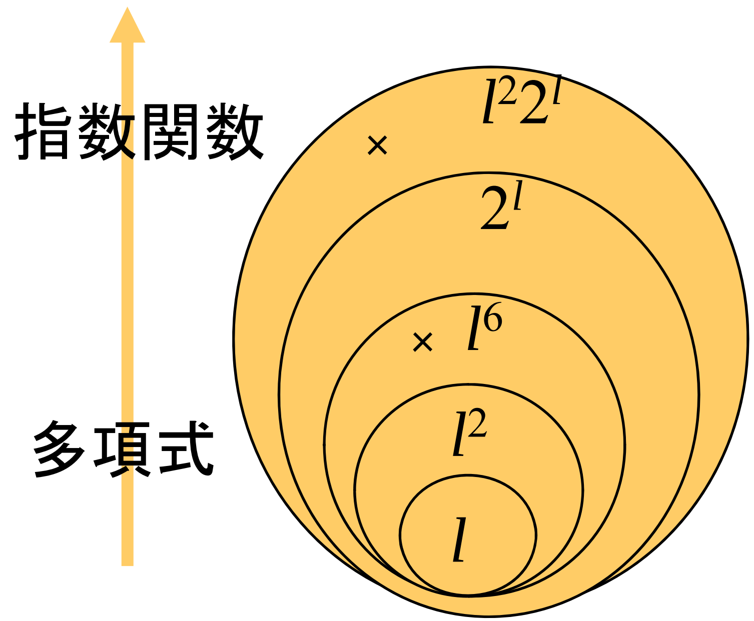
自然数上の関数 t に対し，時間計算量が $O(t)$ となる集合（i.e.,認識問題）の全体を **$O(t)$ 時間計算量クラス**といい，そのクラスを**TIME(t)**と表す.

また，t のような関数を制限時間と呼ぶ.

たとえば，$O(l^2 2^l)$ 時間で認識可能な集合を集めたクラスがTIME($l^2 2^l$)であり，集合 PRIME はその一要素.

$$\mathrm{PRIME} \in \mathrm{TIME}(l^2 2^l)$$

今では PRIME $\in$ TIME (l^6)



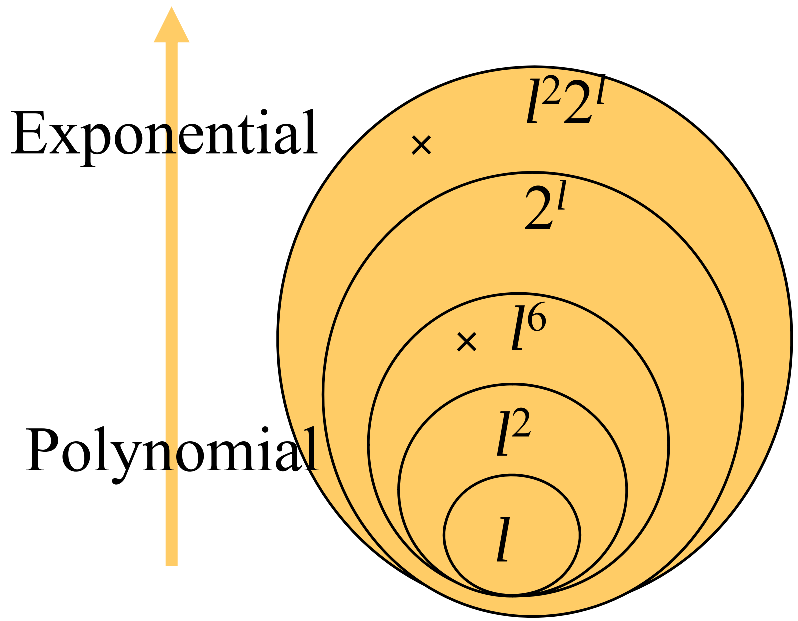
Def.4.5.

For a function t over natural numbers, the set of all sets (i.e. recognition problems) with time complexities $O(t)$ is called ***O*(*t*)-time complexity class**, and it is denoted by TIME(t). And such a function t is called a time limit.

For example, a class of sets recognizable in time $O(l^2 2^l)$ is TIME($l^2 2^l$), and the set PRIME is one element.

$$\text{PRIME} \in \text{TIME}(l^2 2^l)$$

Now, PRIME $\in$ TIME (l^6)



第5章 代表的な計算量クラス

5.1. 代表的な時間計算量クラス

$$\mathcal{P} \equiv \bigcup_{p:\text{多項式}} \mathrm{TIME}(p(l))$$

$$\mathcal{E} \equiv \bigcup_{c>1} \mathrm{TIME}(2^{cl})$$

$$\mathcal{EXP} \equiv \bigcup_{p:\text{多項式}} \mathrm{TIME}(2^{p(l)})$$

$\mathcal{C}$集合：　計算量クラス$\mathcal{C}$に入る集合.
$\mathcal{C}$問題：　$\mathcal{C}$集合の認識問題

ある問題が$\mathcal{P}$に入っていないなら、
現実的には手に負えない…

Chapter 5
Representative Complexity Classes

5.1. Representative time complexity classes

$$\mathcal{P} \equiv \bigcup_{p:\mathrm{polynomial}} \mathrm{TIME}(\mathrm{p}(l))$$

$$\mathcal{E} \equiv \bigcup_{c>1} \mathrm{TIME}(2^{cl})$$

$$\mathcal{EXP} \equiv \bigcup_{p:\mathrm{polynomial}} \mathrm{TIME}(2^{p(l)})$$

$\mathcal{C}$ set : set in the complexity class $\mathcal{C}$.
$\mathcal{C}$ problem : problem of recognizing a $\mathcal{C}$ set.

Problems not in $\mathcal{P}$ are intractable
from the practical viewpoint…

例5.1: クラス$\mathcal{P}$, $\mathcal{E}$, $\mathcal{EXP}$では，多項式時間程度の違いは問題ではない．

$\mathcal{P}$: 多項式 × 多項式→多項式
$\mathcal{E}$: 2の線形乗 × 多項式 → 2の線形乗
$\mathcal{EXP}$: 2の多項式乗 × 多項式 → 2の多項式乗

例5.2: PRIMEの計算量クラス
例4.7 → PRIME $\in$ TIME(2^l)
故に，PRIME $\in$ $\mathcal{E}$

余談: 2002年に $O(l^6)$ のアルゴリズムが考案されたので、今では$\mathcal{P}$

定義5.1. T: 制限時間の集合

$\bigcup_{t\in T}$TIME(t): T時間計算量クラス
→これをTIME(T)と表す．

定理5.1: (1) $\mathcal{P} = \cup_{c>0}$TIME(l^c), (2) $\mathcal{EXP} = \cup_{c>0}$ TIME(2^{l^c})

Ex.5.1: Polynomial makes no serious difference in the classes $\mathcal{P}$, $\mathcal{E}$, $\mathcal{EXP}$.

$\mathcal{P}$: polynomial × polynomial → polynomial

$\mathcal{E}$: linear power of 2 × polynomial → linear power of 2

$\mathcal{EXP}$: poly. power of 2 × poly. → poly. power of 2

Ex.5.2: Complexity class of PRIME

Ex.4.7 → PRIME $\in$ TIME(2^l)

Thus, PRIME $\in \mathcal{E}$

$O(l^6)$ time algorithm puts it into $\mathcal{P}$!!

Def.5.1: $\mathcal{T}$: set of time limits

$\bigcup_{t \in \mathcal{T}}$ TIME(t): $\mathcal{T}$ time complexity class

→It is denoted by TIME($\mathcal{T}$).

Theorem5.1 (1) $\mathcal{P} = \cup_{c>0}$ TIME(l^c), (2) $\mathcal{EXP} = \cup_{c>0}$ TIME(2^{l^c})

定理5.1: (1) $\mathcal{P} = \cup_{c>0}\mathrm{TIME}(l^c)$,　(2) $\mathcal{EXP} = \cup_{c>0}\mathrm{TIME}(2^{l^c})$

証明: (2)の証明は省略.
T_1: l^cという形の多項式の集合.
T_2: 多項式の全体
→ $T_1 \subseteq T_2$なので, $\mathrm{TIME}(T_1) \subseteq \mathrm{TIME}(T_2)$
p: 任意の多項式 (pはT_2の任意の要素)
多項式pの最大次数をkとすると, $p(l) = O(l^k)$
定理4.3より,
$\mathrm{TIME}(p(l)) \subseteq \mathrm{TIME}(l^k) \subseteq \mathrm{TIME}(T_1)$
したがって, $\mathrm{TIME}(T_1) = \mathrm{TIME}(T_2)$

証明終

定理4.3:
すべての制限時間 t_1,t_2 に対し、
$t_1=O(t_2)$ ならば $\mathrm{TIME}(t_1)\subseteq\mathrm{TIME}(t_2)$

Theorem 5.1: (1) $\mathcal{P} = \cup_{c>0}$ TIME(l^c), (2) $\mathcal{EXP} = \cup_{c>0}$TIME($2^{l^c}$)

Proof: The proof of (2) is omitted.

$\mathcal{T}_1$: set of polynomials of the form of l^c.

$\mathcal{T}_2$: set of all polynomials

→ since $\mathcal{T}_1 \subseteq \mathcal{T}_2$, TIME($\mathcal{T}_1$) $\subseteq$ TIME($\mathcal{T}_2$)

p: arbitrary polynomial (p is any element of $\mathcal{T}_2$)

if the maximum degree of a polynomial p is k, $p(l) = O(l^k)$

From Theorem 4.3,

TIME($p(l)$) $\subseteq$ TIME(l^k) $\subseteq$ TIME($\mathcal{T}_1$)

Therefore, TIME($\mathcal{T}_1$) = TIME($\mathcal{T}_2$)

Q.E.D.

Theorem 4.3:

For any times t_1,t_2,

t_1=O(t_2) implies TIME(t_1)$\subseteq$TIME(t_2)

例5.3. 命題論理式評価問題(PROP-EVAL)

入力: $<F, <a_1, a_2, \dots, a_n>>$

Fは拡張命題論理式 $\wedge \vee \neg \rightarrow \leftrightarrow$

$(a_1, a_2, \dots, a_n)$は F に対する真理値割り当て

質問: $F(a_1, a_2, \dots, a_n) = 1$?

(x,y)	$x \rightarrow y$ $(\neg x \vee y)$	$x \leftrightarrow y$ $((x \rightarrow y) \wedge (y \rightarrow x))$
(0,0)	1	1
(0,1)	1	0
(1,0)	0	0
(1,1)	1	1

Ex.5.3. Problem of evaluating propositional expression(PROP-EVAL)

Input : $<F, < a_1, a_2, \ldots , a_n >>$

F is an extended prop. expression

$(a_1, a_2, \ldots , a_n)$ is a truth assignment to F

Question : $F(a_1, a_2, \ldots , a_n) = 1$?

(x,y)	$x \rightarrow y$ $(\neg x \vee y)$	$x \leftrightarrow y$ $((x \rightarrow y) \wedge (y \rightarrow x))$
(0,0)	1	1
(0,1)	1	0
(1,0)	0	0
(1,1)	1	1

例5.3. 命題論理式評価問題(PROP-EVAL)

入力: $<F, < a_1, a_2, \ldots , a_n >>$

Fは拡張命題論理式 $\wedge \vee \neg \rightarrow \leftrightarrow$

$(a_1, a_2, \ldots , a_n)$はFに対する真理値割り当て

質問: $F(a_1, a_2, \ldots , a_n) = 1$?

拡張命題論理式 F がコード化されたもの $\lceil F \rceil$ から計算木を作る.

計算木は$O(|\lceil F \rceil|^3)$時間で構成できる.

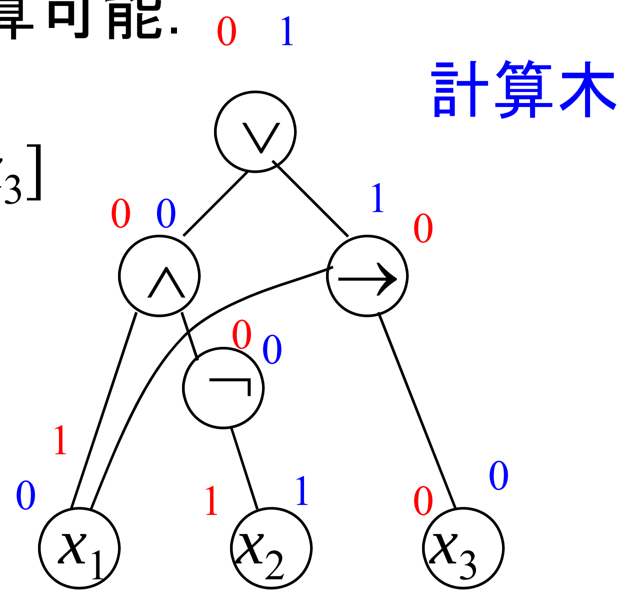
計算木が得られていれば, ボトムアップ式で

$F(a_1, a_2, \ldots , a_n)$ の値は容易に計算可能.

例: $F(x_1, x_2, x_3) = [x_1 \wedge \neg x_2] \vee [x_1 \rightarrow x_3]$

$F(0,1,0)=1$

$F(1,1,0)=0$



よって PROP-EVAL $\in \mathcal{P}$

Ex.5.3. Problem of evaluating propositional expression(PROP-EVAL)

Input: $<F, < a_1, a_2, \ldots , a_n >>$

F is an extended prop. expression

$(a_1, a_2, \ldots , a_n)$ is a truth assignment to F

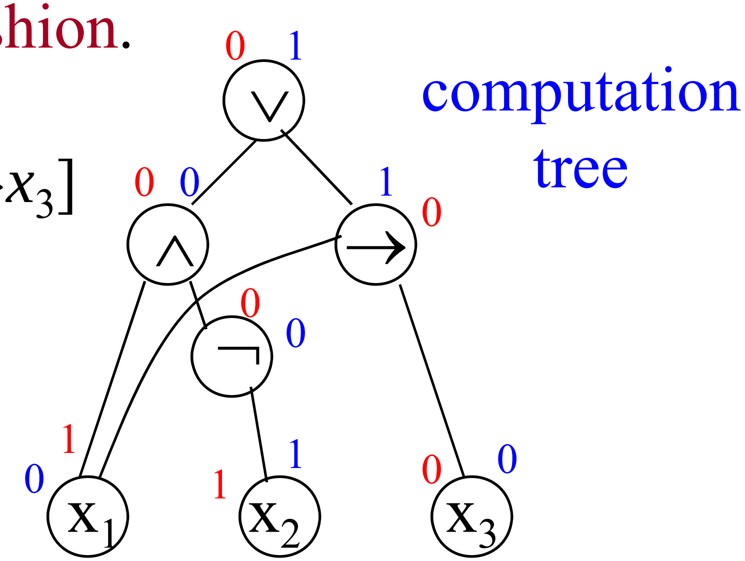
Question: $F(a_1, a_2, \ldots , a_n) = 1$?

Construct a computation tree from a code $\lceil F \rceil$ of ext. prop. expression

It is built in time $O(|\lceil F \rceil|^3)$.

If computation tree is available, we can easily obtain the value $F(a_1, a_2, \ldots , a_n)$ in a bottom-up fashion.

Ex.: $F(x_1, x_2, x_3) = [x_1 \wedge \neg x_2] \vee [x_1 \rightarrow x_3]$



$F(0,1,0)=1$

$F(1,1,0)=0$

Hence PROP-EVAL $\in \mathcal{P}$

例5.3. 命題論理式充足性問題:2和積形(2SAT)

入力: $<F>$ Fは2和積形命題論理式

質問: $F(a_1, a_2, \ldots, a_n) = 1$を満たす割り当てがあるか?

和積形:

$F = (\bullet \lor \bullet \lor \ldots \lor \bullet) \land (\bullet \lor \ldots \lor \bullet) \land \ldots \land (\ldots)$

- リテラルの論理<u>和</u>の論理<u>積</u>で表現されたもの

<u>ちょうど</u>/たかだか

k和積形(k SAT)

- 和積形の各論理和が k 個のリテラルを含む

- 3SAT, 4SAT も同様に定義できる。
- SAT: 各論理和のリテラルの個数に制限がないもの
- ExSAT: 入力が拡張命題論理式(→や ↔も許す)

Ex. 5.3. 2-Satisfiability (2SAT)

Input : $<F>$ F is 2-conjunctive normal form

Question : Is there any assignment such that $F(a_1, a_2, \ldots, a_n) = 1$?

Conjunctive Normal Form (CNF)

$$F = (\bullet \vee \bullet \vee \ldots \vee \bullet) \wedge (\bullet \vee \ldots \vee \bullet) \wedge \ldots \wedge (\ldots)$$

- described by $\wedge$ of $\vee$ of literals.

exactly/at most

k SAT

- Each closure contains k literals

- We can define 3SAT, 4SAT similarly.
- SAT consists of any CNF.
- ExSAT consists of any extended propositional expression.

例5.4: 到達可能性問題(ST-CON)

入力 : <G,s,t> : 無向グラフG, $1 \leqq s,t \leqq n(=|G|)$

質問 : G上で s から t への道があるか?

- 閉路とは、始点と終点が同じである路
- オイラー閉路とは、すべての辺を一度づつ通る閉路
- ハミルトン閉路とは、すべての頂点を一度づつ通る閉路

例5.4: 一筆書き閉路問題(DEULER)

入力 : <G>: 有向グラフG

質問 : Gはオイラー閉路をもつか?

例5.5: ハミルトン閉路問題(DHAM)

入力 : <G>: 有向グラフG

質問 : Gはハミルトン閉路をもつか?

Ex. 5.4: Graph reachability problem (ST-CON)

Input : <G,s,t> : an undirectd graph G, $1 \leqq s,t \leqq n(=|G|)$

Question : Does G have a path from s to t?

- Cycle is a path that shares two endpoints.
- Euler cycle is a cycle that visits all edges once.
- Hamiltonian cycle is a cycle that visits all vertices once.

Ex. 5.4: Euler cycle problem (DEULER)

Input : <G>: a directed graph G

Question : Does G have an Euler cycle?

Ex. 5.5 Hamiltonian cycle problem (DHAM)

Input : <G>: a directed graph G

Question : Does G have a Hamiltonian cycle?

以下の事実が知られている:

- 以下の問題は $\mathcal{P}$ に属する:
  - ✓ PROP-EVAL, 2SAT, ST-CON, DEULER
- 以下の問題は $\mathcal{E}$ に属する、が、、、
  - ✓ 3SAT, DHAM

$\mathcal{P}$ と $\mathcal{E}$ の間(?)のクラス $\mathcal{NP}$

It is known that:

- The following problems are in $\mathcal{P}$:
  - ✓ PROP-EVAL, 2SAT, ST-CON, DEULER
- The following problems are in $\mathcal{E}$, but…
  - ✓ 3SAT, DHAM

The class $\mathcal{NP}$ between $\mathcal{P}$ and $\mathcal{E}$?

5.2. クラス$\mathcal{NP}$

定義5.2: 集合 L に対して次の条件を満たす多項式 q と
多項式時間計算可能述語 R が存在したとする.

$$各x \in \Sigma^* で x \in L \leftrightarrow \exists w \in \Sigma^* : |w| \le q(|x|)[R(x,w)] \quad (5.1)$$

つまり, $L = \{x : \exists w \in \Sigma^* [\, |w| \le q(|x|) \wedge R(x,w)]\}$

このとき, Lを$\mathcal{NP}$集合といい, Lの認識問題を$\mathcal{NP}$問題という.
また, $\mathcal{NP}$集合の全体をクラス$\mathcal{NP}$という.

補注: 各 $x \in \Sigma^*$に対して, 論理式 $|w| \le q(|x|) \wedge R(x,w)$
を満たす $w_x \in \Sigma^*$をxの(多項式長の)証拠という.
以下では, $\exists w \in \Sigma^* : |w| \le q(|x|) \quad \Rightarrow \quad \exists_q w$ と略記.

「入力サイズの多項式長の証拠が与えられたとき, これが問題の条件を満たすかどうかを多項式時間で判定できる. 」

補足: $\mathcal{NP}$=**N**ondeterministic **P**olynomial

5.2. Class $\mathcal{NP}$

Def. 5.2: Suppose that we have a polynomial q and polynomial time computable predicate R for a set L such that

$$\text{for each } x \in \Sigma^*, x \in L \leftrightarrow \exists w \in \Sigma^* :|w| \le q(|x|)[R(x,w)]$$

i.e., $L = \{x : \exists w \in \Sigma^* [\, |w| \le q(|x|) \wedge R(x,w)]\}$ (5.1)

Then, L is called an $\mathcal{NP}$ set, and the problem of recognizing L is called an $\mathcal{NP}$ problem.
Also, the whole set of $\mathcal{NP}$ sets is called the class $\mathcal{NP}$.

Note: For each $x \in \Sigma^*$, $w_x \in \Sigma^*$ satisfying the predicate $|w| \le q(|x|) \wedge R(x,w)$ is called (polynomial) witness of x.
Hereafter, we use notation $\exists w \in \Sigma^* :|w| \le q(|x|) \quad \Rightarrow \quad \exists_q w$

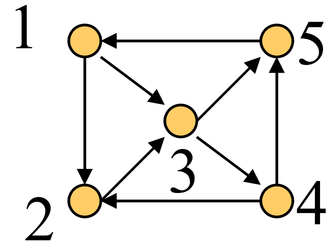
"Given a witness of polynomial length in the input size, we can determine in polynomial time whether it satisfies the condition of a given problem."

$c.f.:\mathcal{NP}$=**N**ondeterministic **P**olynomial

例5.7: ハミルトン閉路問題 (DHAM) $\in \mathcal{NP}$

グラフの頂点は1〜n と番号づけされていると仮定.
ハミルトン閉路の辿り方➔ 1〜n の順列$< l_1, l_2, \ldots , l_n >$
この順列が多項式長の証拠

(注)全部で $n!$〜n^n 通りある

例: 

証拠の候補
<1,2,3,4,5> ➔ハミルトン閉路 ➔証拠
<1,2,3,5,4> ➔ハミルトン閉路でない
<1,4,3,2,5> ➔ハミルトン閉路でない

$R_D(x, w)$ ↔[xはあるグラフG(n頂点)のコード]
∧ [wは1〜nの順列$< l_1, l_2, \ldots , l_n >$]
∧[wはGのハミルトン閉路を表している]

すべての $x \in \Sigma^*$について次の関係が成り立つ.
xがあるグラフGのコードになっているとき:

$$x \in \mathrm{DHAM} \leftrightarrow \exists w_G (=< l_1, \ldots, l_n >)[R_D(x, w_G)]$$

xがグラフのコードになっていないとき: $\forall w[\neg R_D(x, w)]$

Ex.5.7: Hamilton Cycle Problem (DHAM) ∈ $\mathcal{NP}$

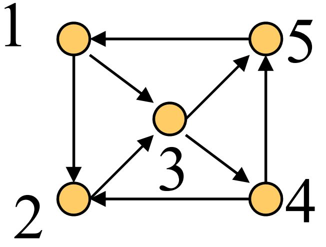
Assume graph vertices are numbered $1 \sim n$.

Trace on a Hamilton cycle ➔ permutation of $1 \sim n$ $< l_1, l_2, \ldots, l_n >$

This permutation is a witness of polynomial length.

(c.f.)There are $n! \sim n^n$ many

Ex.:



candidates of witness

<1,2,3,4,5> ➔ Hamilton cycle ➔ witness

<1,2,3,5,4> ➔ not Hamilton cycle

<1,4,3,2,5> ➔ not Hamilton cycle

$R_D(x, w) \leftrightarrow$ [x is a code of a graph G(with n vertices)]

$\wedge$ [w is a permutation of $1 \sim n$: $< l_1, l_2, \ldots, l_n >$]

$\wedge$[w represents a Hamilton cycle in G]

For each $x \in \Sigma^*$ we have

if x is a code of a graph G:

$$x \in \mathrm{DHAM} \leftrightarrow \exists w_G (=< l_1, \ldots, l_n >)[R_D(x, w_G)]$$

if x is not a code of any graph: $\forall w[\neg R_D(x, w)]$

例5.8: 命題論理式充足性問題(3SAT, SAT, ExSATなど)

目標 : ExSAT $\in \mathcal{NP}$

$F(x_1, \ldots, x_n)$: 任意の拡張命題論理式

Fが充足可能 $\leftrightarrow$ ∃$a_1, \ldots, a_n$: 各a_iは1か0 [$F(a_1, \ldots, a_n) = 1$]

証拠の長さq_E

Fへの真偽値の割り当てを$< a_1, \ldots, a_n >$で表す.

→ 長さは $3(n+n+1)=6n+3 \leq 6|\ulcorner F \urcorner| + 3$

$q_E(l) = 6l+3$

述語R_E

$R_E(x, w) \leftrightarrow$ [xはある拡張命題論理式F (n変数)のコード]

$\wedge$ [wはFへの割り当て$< a_1, a_2, \ldots, a_n >$]

$\wedge$ [$F(a_1, \ldots, a_n) = 1$]

計算木を用いると$F(a_1, \ldots, a_n)$ の値は多項式時間で計算可能.

よって, R_Eも多項式時間で計算可能.

Ex.5.8: Satisfiability Problem of Prop. Express. (3SAT, SAT, ExSAT)

Goal : ExSAT $\in \mathcal{NP}$

$F(x_1, \dots, x_n)$: arbitrary extended prop. logic. expression

F is satisfiable $\leftrightarrow$ $\exists a_1, \dots, a_n$: each a_i is 0 or 1 $[F(a_1, \dots, a_n) = 1]$

length of a witness q_E

Truth assignment to F is denoted by $< a_1, \dots, a_n >$.

→ its length is $3(n+n+1) = 6n+3 \leq 6|\lceil F \rceil| + 3$

$q_E(l) = 6l+3$

predicate R_E

$R_E(x, w) \leftrightarrow [x$ is a code of an extended prop. express. F (n variables)$]$

$\wedge$ $[w$ is an assignment to $F : < a_1, a_2, \dots, a_n >]$

$\wedge$ $[F(a_1, \dots, a_n) = 1]$

Using a computation tree, the value of $F(a_1, \dots, a_n)$ is computed in polynomial time. Thus, R_E is also computable in polynomial time.

$\mathcal{NP}$集合であることの意味は何か?

(5.1)を満たすq, Rを用いると, $x \in L$? を次のように判定できる.

for each $w \in \Sigma^{\le q(|x|)}$ do
  if $R(x, w)$ then accept end-if
end-for;
reject;

長さが$q(|x|)$以下の文字列をすべて列挙して調べれば,
acceptかrejectか判定できる. ただ, そのような文字列は
2の$q(|x|)$乗個(指数関数)存在することに注意.

上記の計算方式で認識できる集合を$\mathcal{NP}$集合と考えてよい.

What does it mean by being an $\mathcal{NP}$ set?

Using q and R satisfying the predicate characterizing an $\mathcal{NP}$ set, we can determine $x \in L$? in the following way.

```
for each w ∈ Σ^{≤q(|x|)} do
  if R(x, w) then accept end-if
end-for;
reject;
```

If we enumerate and check all possible strings of length at most $q(|x|)$, then we can accept or reject them. Here note that there are 2 to the $q(|x|)$ (exponentially many) such strings.

We may think that those sets recognizable as above are $\mathcal{NP}$ sets.

$\mathcal{NP}$に関連したクラス

定義5.3. 集合Lは，その補集合$\overline{L}$が$\mathcal{NP}$に属しているとき，
co-$\mathcal{NP}$集合という．また，co-$\mathcal{NP}$集合の全体を**クラスco-$\mathcal{NP}$**という．

補注: co-$\mathcal{P}$ を定義しても $\mathcal{P}$ と同じなので無意味.

定理5.5. すべての集合 L に対し，次の条件は同値.
(a) $L \in$ co-$\mathcal{NP}$
(b) 集合 L を，適当な多項式 q と多項式時間計算可能述語Qを用いて，

$$L = \{x : \forall w \in \Sigma^* : |w| \le q(|x|)[Q(x,w)]\}$$

と表せる.

Classes related to $\mathcal{NP}$

Def.5.3. A set L is called a **co-$\mathcal{NP}$** set if its complement $\overline{L}$ belongs to $\mathcal{NP}$. The whole family of co-$\mathcal{NP}$ sets is called the class **co-$\mathcal{NP}$**.

Note: It is nonsense to define co-$\mathcal{P}$ since it is equal to $\mathcal{P}$.

Theorem 5.5. For every set L, the following conditions are equivalent.

(a) $L \in$ co-$\mathcal{NP}$

(b) The set L can be represented as

$$L = \{x : \forall w \in \Sigma^* : |w| \leq q(|x|)[Q(x, w)]\}$$

by using some polynomial q and polynomial-time computable predicate Q.

例5.9: 素数判定問題

$$\lceil n \rceil \notin \mathrm{PRIME} \leftrightarrow \exists m{:}1 < m < n[\ n \bmod m = 0]$$

したがって，$q_p(n) = n$とし，

$$R_p(x, w) \leftrightarrow [x \notin \mathsf{N}] \vee [[w \in \mathsf{N}] \wedge [1 < m < n] \wedge [n \ \mathsf{mod}\ m = 0]]$$

(ただし，n, mは各々x, wが表す自然数，
Nは自然数の2進表記全体)

と定義すると，

すべての$x \in \Sigma^*$に対し，　$x \notin \mathrm{PRIME} \leftrightarrow \exists q_p w[R_p(x, w)]$

これは，$x \notin \mathrm{PRIME}$に対する証拠

よって，$\overline{\mathrm{PRIME}} \in \mathcal{NP}$, i.e., $\mathrm{PRIME} \in$ co-$\mathcal{NP}$

実際，$Q(x, w) \leftrightarrow \neg R_p(x, w)$ とすると

$$\mathrm{PRIME} = \{x{:}\ \forall q_p w\ [Q_p(x, w)]\}$$

と表せる．

$\mathrm{PRIME} \in \mathcal{NP}$ も示せるが，その証明はもっと複雑．

Ex.5.9: Primality testing

$$\lceil n \rceil \notin \mathsf{PRIME} \leftrightarrow \exists m : 1 < m < n[\ n \ \mathsf{mod}\ m = 0]$$

Therefore, for $q_p(n) = n$,

$$R_p(x, w) \leftrightarrow [x \notin \mathsf{N}] \vee [[w \in \mathsf{N}] \wedge [1 < m < n] \wedge [n \ \mathsf{mod}\ m = 0]]$$

(where, n and m are natural numbers represented by x and w. N is a set of all natural numbers in the binary form)

This definition leads to

for every $x \in \Sigma^*$ we have $x \notin \text{PRIME} \leftrightarrow \exists q_p w[R_p(x, w)]$

This is a witness to $x \notin$ PRIME

Thus, $\overline{\text{PRIME}} \in \mathcal{NP}$, i.e., PRIME $\in$ co-$\mathcal{NP}$

In fact, using $Q(x, w) \leftrightarrow \neg R_p(x, w)$, PRIME can be expressed as

PRIME = $\{x: \forall q_p w\ [Q_p(x, w)]\}$

We can also show that PRIME $\in \mathcal{NP}$, but its proof is more complex.

$\mathcal{NP}$問題の例

・**合成数判定問題**(COMPOSITE)

入力：自然数n

質問：nは合成数か？（素数でないか？）

・**ナップサック問題**(KNAP)

入力：自然数の組$< a_1, a_2, \ldots, a_n, b >$

質問：$\sum_{i \in S} a_i = b$ となる添字の集合 $S \subseteq \{1, \ldots, n\}$があるか？

・**箱詰め問題**(BIN)

入力：自然数の組$< a_1, a_2, \ldots, a_n, b, k>$

質問：添字の集合$U=\{1, \ldots, n\}$を$U_1, \ldots, U_k$のk個に分割し，各jで $\sum_{i \in U_j} a_i \le b$ とすることは可能か？

・**頂点被覆問題**(VC)

入力：無向グラフGと自然数kの組$<G, k>$

質問：Gにk頂点の頂点被覆が存在するか？

頂点被覆S:
どの辺(u,v)も
u,vの一方は
Sに含まれる

Examples of $\mathcal{NP}$ problems

- **Composite Number Testing Problem**(COMPOSITE)
  input : natural number n
  question : Is n composite? (Is it not prime?)

- **Knapsack Problem**(KNAP)
  input : n+1 tuple of natural numbers $< a_1, a_2, \ldots, a_n, b >$
  question : Is there a set of indices $S \subseteq \{1, \ldots, n\}$ s.t. $\sum_{i \in S} a_i = b$?

- **Bin Packing Problem**(BIN)
  input : n+2 tuple of natural numbers $< a_1, a_2, \ldots, a_n, b, k >$
  question : Is there a partition of a set of indices $U=\{1, \ldots, n\}$ into $U_1, \ldots, U_k$ such that $\sum_{i \in U_j} a_i \le b$ for each j?

- **Vertex Cover Problem**(VC)
  input : pair of undirected graph G and natural number k $<G, k>$
  question : Is there a vertex cover of k vertices over G?

Vertex Cover S contains at least one of u and v for each edge (u,v).

5.3. 計算量クラス間の関係

定理5.6: $\mathcal{P} \subseteq \mathcal{E} \subseteq \mathcal{EXP}$.

定義より，明らか.

定理5.7: $\mathcal{P} \subsetneq \mathcal{E} \subsetneq \mathcal{EXP}$.

階層定理(定理4.4):
任意の制限時間 t_1, t_2 に対し、

$$\forall c > 0, \overset{\infty}{\forall} n[ct_1(n)^2 \leq t_2(n)) \rightarrow \mathrm{TIME}(t_1) \underset{\neq}{\subset} \mathrm{TIME}(t_2)$$

証明:

(1) $\mathcal{P} \subsetneq \mathcal{E}$.

$t_1(n)=2^n$, $t_2(n)=2^{3n}$とすると，<u>階層定理</u>より，

$$\mathrm{TIME}(2^n) \subsetneq \mathrm{TIME}(2^{3n})$$

一方，$\mathcal{P} \subseteq \mathrm{TIME}(2^n) \subsetneq \mathrm{TIME}(2^{3n}) \subseteq \mathcal{E}$だから，

$$\mathcal{P} \subsetneq \mathcal{E}.$$

(2)も同様.

証明終

5.3. Relation in the Complexity Class

Theorem 5.6: $\mathcal{P} \subseteq \mathcal{E} \subseteq \mathcal{EXP}$.

Obvious from the definition.

Theorem 5.7: $\mathcal{P} \subsetneqq \mathcal{E} \subsetneqq \mathcal{EXP}$.

Hierarchy Thm. (Thm. 4.4):
For any times t_1, t_2 ,

$$\forall c > 0, \overset{\infty}{\forall} n[ct_1(n)^2 \le t_2(n)) \to \mathrm{TIME}(t_1) \underset{\neq}{\subset} \mathrm{TIME}(t_2)$$

Proof:

(1) $\mathcal{P} \subsetneqq \mathcal{E}$.

For $t_1(n)=2^n$, $t_2(n)=2^{3n}$, from the hierarchy theorem we have

$$\mathrm{TIME}(2^n) \subsetneqq \mathrm{TIME}(2^{3n})$$

On the other hand, since $\mathcal{P} \subseteq \mathrm{TIME}(2^n) \subsetneqq \mathrm{TIME}(2^{3n}) \subseteq \mathcal{E}$

$$\mathcal{P} \subsetneqq \mathcal{E}.$$

(2) is similar.

Q.E.D.

定理5.8.

(1) $\mathcal{P} \subseteq \mathcal{NP}$, $\mathcal{P} \subseteq$ co-$\mathcal{NP}$ (よって, $\mathcal{P} \subseteq \mathcal{NP} \cap$ co-$\mathcal{NP}$)

(2) $\mathcal{NP} \subseteq \mathcal{EXP}$, co-$\mathcal{NP} \subseteq \mathcal{EXP}$ (よって, $\mathcal{NP} \cup$ co-$\mathcal{NP} \subseteq \mathcal{EXP}$)

証明: (1) $\mathcal{P} \subseteq \mathcal{NP}$（$\mathcal{P} \subseteq$ co-$\mathcal{NP}$ も同様）

L: 任意の$\mathcal{P}$集合

➔ Lは多項式時間で認識可能

よって, 多項式時間計算可能述語Pを用いて次のように書ける.

$\forall x \in \Sigma^* : [x \in L \leftrightarrow P(x)]$ or $P = \{x: P(x)\}$

$R(x, w) = P(x)$と定義（第2引数は無視）

→ 任意の多項式qについて,

$L = \{x: \exists_q w[\, R(x,w)]\}$

よって, $\mathcal{NP}$の定義より, $L \in \mathcal{NP}$ i.e., $\mathcal{P} \subseteq \mathcal{NP}$.

Theorem 5.8.
(1) $\mathcal{P} \subseteq \mathcal{NP}$, $\mathcal{P} \subseteq$ co-$\mathcal{NP}$ (thus, $\mathcal{P} \subseteq \mathcal{NP} \cap$ co-$\mathcal{NP}$)
(2) $\mathcal{NP} \subseteq \mathcal{EXP}$, co-$\mathcal{NP} \subseteq \mathcal{EXP}$ (thus, $\mathcal{NP} \cup$ co-$\mathcal{NP} \subseteq \mathcal{EXP}$)

Proof:

(1) $\mathcal{P} \subseteq \mathcal{NP}$ ($\mathcal{P} \subseteq$ co-$\mathcal{NP}$ is similar)

L: arbitrary $\mathcal{P}$ set

➔ L is recognizable in polynomial time

Thus, we have the following description using a polynomial-time computable predicate P.

$\forall x \in \Sigma^* : [x \in L \leftrightarrow P(x)]$ or $P = \{x : P(x)\}$

We define $R(x, w) = P(x)$ (neglecting the second argument)

→ for any polynomial q,

$L = \{x : \exists_q w[\, R(x,w)]\}$

Thus, from the definition of $\mathcal{NP}$, $L \in \mathcal{NP}$ i.e., $\mathcal{P} \subseteq \mathcal{NP}$.

(2) $\mathcal{NP} \subseteq \mathcal{EXP}$ (co-$\mathcal{NP} \subseteq \mathcal{EXP}$)

L: 任意の$\mathcal{NP}$集合

➔多項式qと多項式時間計算可能述語Rが存在して，

$$L = \{x : \exists_q w[R(x,w)]\} = \{x : \exists_q w[|w| \le q(|x|) \wedge R(x,w)]\}$$

qとRを用いて，Lを認識するプログラムを作る．

```
prog L(input x);
begin
    for each  w ∈ Σ^{≤q(|x|)} do
      if R(x, w) then accept end-if
    end-for;
    reject
end.
```

長さlの入力に対するプログラムの時間計算量:

Rは多項式時間計算可能だったから，ある多項式pに対し，

Rの計算時間$=p(|x| + |w|) \le p(l + q(l)) \longleftarrow l$ の多項式

全体では，$\{p(l+q(l)) + cq(l)\}2^{q(l)} + d = O(2^{l+q(l)})$

よって，$L \in \mathcal{EXP}$ → $\mathcal{NP} \subseteq \mathcal{EXP}$ 証明終

(2) $\mathcal{NP} \subseteq \mathcal{EXP}$ (co-$\mathcal{NP} \subseteq \mathcal{EXP}$)

L: any $\mathcal{NP}$ set

➔There is some polynomial q and polynomial-time computable predicate R such that

$$L = \{x : \exists_q w[R(x,w)]\} = \{x : \exists_q w[|w| \le q(|x|) \wedge R(x,w)]\}$$

program recognizing L using q and R

```
prog L(input x);
begin
    for each w ∈ Σ^{≤q(|x|)} do
      if R(x, w) then accept end-if
    end-for;
    reject
end.
```

time complexity of the program for an input of length *l*:

Since R is polynomial-time computable, for some polynomial q

time of $R = p(|x| + |w|) \le p(l + q(l))$ ⟵ polynomial of l

In total, $\{p(l+q(l)) + cq(l)\}2^{q(l)} + d = O(2^{l+q(l)})$

Hence, $L \in \mathcal{EXP}$ ➔ $\mathcal{NP} \subseteq \mathcal{EXP}$ Q.E.D.

定理5.9.
(1) $\mathcal{NP} \subseteq$ co-$\mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$
(2) co-$\mathcal{NP} \subseteq \mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$
(3) $\mathcal{NP} \neq$ co-$\mathcal{NP} \rightarrow \mathcal{P} \neq \mathcal{NP}$.

補注: (3)より，$\mathcal{NP} \neq$co-$\mathcal{NP}$の証明は，$\mathcal{P} \neq \mathcal{NP}$の証明より難しい．

証明:　(1) $\mathcal{NP} \subseteq$ co-$\mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$ ((2)の証明も同様）
任意の$L \in$co-$\mathcal{NP}$に対して$L \in \mathcal{NP}$が示せれば，co-$\mathcal{NP} \subseteq \mathcal{NP}$が証明できるので，仮定の$\mathcal{NP} \subseteq$ co-$\mathcal{NP}$と合わせて$\mathcal{NP} =$ co-$\mathcal{NP}$が言える．

$L \in$ co-$\mathcal{NP} \rightarrow \overline{L} \in \mathcal{NP}$ (定義5.3より）
$\rightarrow \overline{L} \in$ co-$\mathcal{NP}$ ($\mathcal{NP} \subseteq$ co-$\mathcal{NP}$より）
$\rightarrow L \in \mathcal{NP}$ (定義5.3と$\overline{\overline{L}}=L$より）

Theorem 5.9

(1) $\mathcal{NP} \subseteq$ co-$\mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$

(2) co-$\mathcal{NP} \subseteq \mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$

(3) $\mathcal{NP} \neq$ co-$\mathcal{NP} \rightarrow \mathcal{P} \neq \mathcal{NP}$.

Note: from (3) the proof for $\mathcal{NP} \neq$ co-$\mathcal{NP}$ is harder than that for $\mathcal{P} \neq \mathcal{NP}$.

Proof: (1) $\mathcal{NP} \subseteq$ co-$\mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$ (proof of (2) is similar)

Since co-$\mathcal{NP} \subseteq \mathcal{NP}$ is shown if we prove $L \in \mathcal{NP}$ for any $L \in$ co-$\mathcal{NP}$

Combining it with the assumption $\mathcal{NP} \subseteq$ co-$\mathcal{NP}$, we have

$\mathcal{NP} =$ co-$\mathcal{NP}$ and so

$L \in$ co-$\mathcal{NP} \rightarrow \overline{L} \in \mathcal{NP}$ (by Definition 5.3)

$\rightarrow \overline{L} \in$ co-$\mathcal{NP}$ ($\mathcal{NP} \subseteq$ co-$\mathcal{NP}$)

$\rightarrow L \in \mathcal{NP}$ (Definition 5.3 and $\overline{\overline{L}}=L$)

(3) $\mathcal{NP} \neq$ co-$\mathcal{NP} \rightarrow \mathcal{P} \neq \mathcal{NP}$.

対偶: $\mathcal{P} = \mathcal{NP} \rightarrow \mathcal{NP} =$ co-$\mathcal{NP}$

$\mathcal{P}$=$\mathcal{NP}$と仮定すると，すべてのLに対し

$L \in \mathcal{NP} \leftrightarrow L \in \mathcal{P}$ （$\mathcal{P} = \mathcal{NP}$ より）

$\leftrightarrow \overline{L} \in \mathcal{P}$ (演習問題5.5)

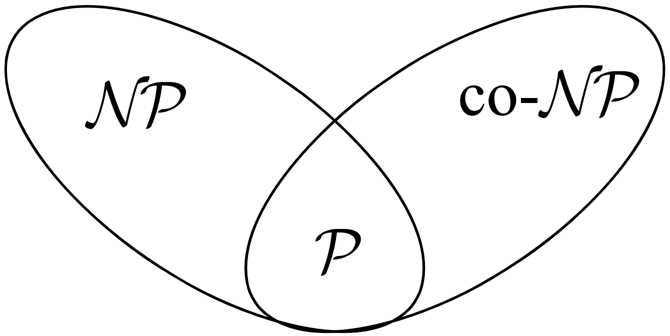
$\leftrightarrow \overline{L} \in \mathcal{NP}$ ($\mathcal{P} = \mathcal{NP}$ より）

$\leftrightarrow L\ (= \overline{L}) \in$ co-$\mathcal{NP}$ (定義5.3より)

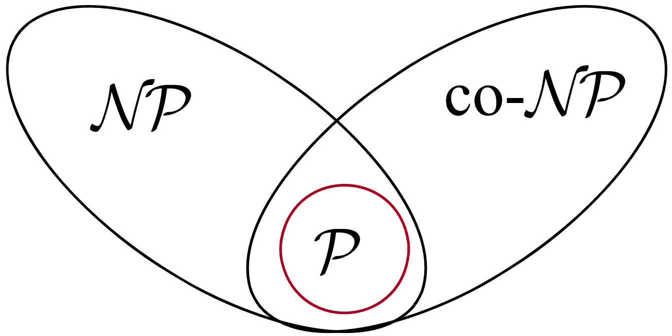
$\therefore \mathcal{NP} =$ co-$\mathcal{NP}$

証明終

$\mathcal{NP} \neq$ co-$\mathcal{NP}$が正しいと



or



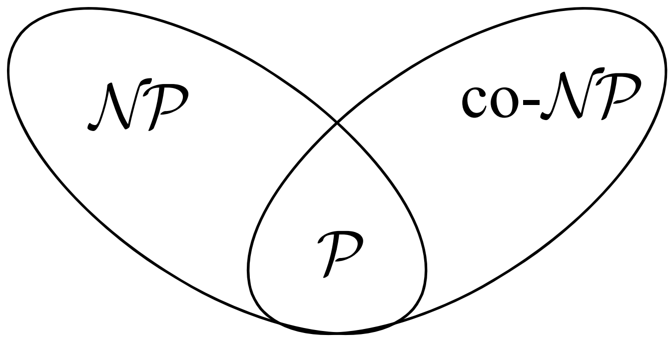
$\boxed{(3)\ \mathcal{NP} \neq \text{co-}\mathcal{NP} \rightarrow \mathcal{P} \neq \mathcal{NP}.}$

$\boxed{\text{Contraposition}: \mathcal{P} = \mathcal{NP} \rightarrow \mathcal{NP} = \text{co-}\mathcal{NP}}$

If we assume $\mathcal{P}=\mathcal{NP}$, for any L we have

$$
\begin{aligned}
L \in \mathcal{NP} &\leftrightarrow L \in \mathcal{P} && (\mathcal{P} = \mathcal{NP}) \\
&\leftrightarrow \overline{L} \in \mathcal{P} && \text{(Exercise 5.5)} \\
&\leftrightarrow \overline{L} \in \mathcal{NP} && (\mathcal{P} = \mathcal{NP}) \\
&\leftrightarrow L\ (= \overline{L}\ ) \in \text{co-}\mathcal{NP} && \text{(Definition 5.3)} \\
&\therefore \mathcal{NP} = \text{co-}\mathcal{NP} && \text{Q.E.D.}
\end{aligned}
$$

If $\mathcal{NP} \neq \text{co-}\mathcal{NP}$ is true,



or