I216 Computational Complexity and Discrete Mathematics

by
Prof. Ryuhei Uehara
and
Prof. Atsuko Miyaji

I216 計算量の理論と離散数学

上原隆平、宮地充子

Computational Complexity

- Goal 1:
  - *"Computable Function/Problem/Language/Set"*

- Goal 2:
  - How can you show "*Difficulty of Problem*"
    - There are *intractable* problems even if they are computable!
      - because they require too many resources (time/space)!
    - Technical terms;

      The class NP, P≠NP conjecture, NP-hardness, reduction

計算量の理論

- ゴール1:
  - *“計算可能な関数/問題/言語/集合”*

- ゴール2:
  - 「問題の困難さ」を示す方法を学ぶ
    - 計算可能な問題であっても、*手におえない*場合がある！
      - 計算に必要な資源（時間・領域）が多すぎるとき
    - 関連する専門用語;

      クラスNP, P≠NP予想, NP困難性, 還元

5. Computational Complexity

5.3. Class NP

5.3.*. Nondeterministic computation

Some problems (like 3SAT, DHAM, etc.) have a *common and natural* property;

- once you get a solution, you can check it efficiently
- without solution, it seems to be quite difficult; you may check all possibilities

• Many natural problems have this property in the real problems.

• This property leads us to the notion of "nondeterministic computation"

5.計算量の理論

5.3.クラスNP

5.3.*. 非決定性計算とは

(3SAT, DHAMといった)ある種の問題には，次のような**共通で自然な**性質がある;

- ひとたび解が得られると，その正当性は簡単にチェックできる
- 解を見つけるのは大変そうに思える．可能な場合をしらみつぶしに調べる必要がありそうに見える．

- 現実の自然な問題の多くはこの性質をもつ．
- この性質を表現するのが「非決定性計算」

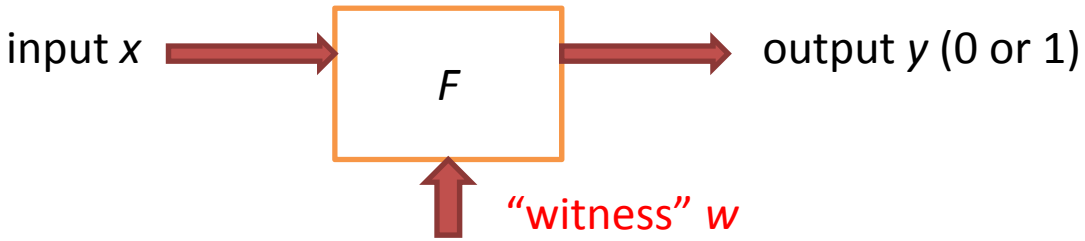
5. Computational Complexity

5.3. Class NP

5.3.*. Nondeterministic computation

"Nondeterministic computation"

- From the viewpoint of Function:



L is called an NP set if there is a function *F* s.t.

1. For each *x*, there is a binary string "witness" *w* s.t.
2. $|w|$ is bounded by a polynomial of $|x|$
3. *F* recognizes $x \in L$ with *w* in polynomial time of $|x|$ and $|w|$

c.f. : NP=**N**ondeterministic **P**olynomial

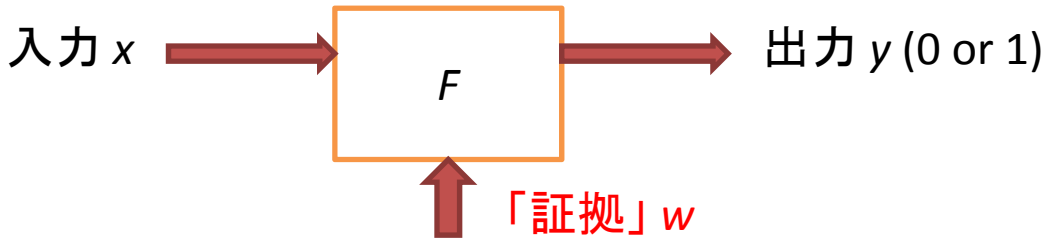
5.計算量の理論

5.3.クラスNP

5.3.*.非決定性計算とは

「非決定性計算」

- 関数の観点からみると:



以下の関数Fが存在するときL はNP 集合と呼ばれる:

1. 各xに対して，2進列の「**証拠**」w が存在
2. $|w|$ は $|x|$の多項式で上から抑えられる
3. F は$|x|$と$|w|$の多項式時間でwを使って$x \in L$ を認識する

c.f.: NP=**N**ondeterministic **P**olynomial

5. Computational Complexity

5.3. Class NP

5.3.*. Nondeterministic computation

"Nondeterministic computation"

- From the viewpoint of Logic:

Such a string w is called "witness"

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

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

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

Then, L is called an NP set, and the problem of recognizing L is called an NP problem.
Also, the whole set of NP sets is called the class NP.

c.f. : NP=**N**ondeterministic **P**olynomial

5.計算量の理論

5.3.クラスNP

5.3.*.非決定性計算とは

「非決定性計算」

- 論理の視点からみると:

集合Lに対して多項式qと多項式で計算できる述語Rがあり,
以下を満たすとする:

この文字列wを「証拠」とよぶ

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

つまり,

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

このとき L はNP 集合とよばれ,
Lの認識問題はNP 問題とよばれる.
また NP 集合全体の集合をクラスNPとよぶ.

c.f. : NP=**N**ondeterministic **P**olynomial

5. Computational Complexity

5.3. Class NP

5.3.*. Nondeterministic computation

A "nondeterministic choice" is a kind of parallel computing that generates two branches.

"Nondeterministic computation"

- From the viewpoint of Turing Machine:

  Suppose that Turing machine has "nondeterministic choice" that admits us to two possible choices at the same time; i.e., it has "one of two cases (0) and (1)" statement.

  - A nondeterministic choice allows to assume of two choices and it will be "*true*" if "*at least one of them is true*".

Then, NP problem L can be recognized by a nondeterministic Turing machine in polynomial time.

c.f. : NP=**N**ondeterministic **P**olynomial

5.計算量の理論

5.3.クラスNP

5.3.*.非決定性計算とは

「非決定性計算」

- チューリングマシンの視点から見ると:

チューリングマシンの「非決定性選択」では，二つの選択肢を「同時に」二つとも選ぶことができる；
つまり「場合(0)と場合(1)のいずれか」という命令がある.

- 非決定性選択は二つの選択肢のうち，「いずれか一方が真」ならば真になる.

「非決定性選択」はある種の並列計算とみなすこともでき，二つの計算プロセスの生成と考えてもよい.

このときNP 問題 Lは，非決定性チューリング機械で多項式時間で受理できる問題.

c.f. : NP=**N**ondeterministic **P**olynomial

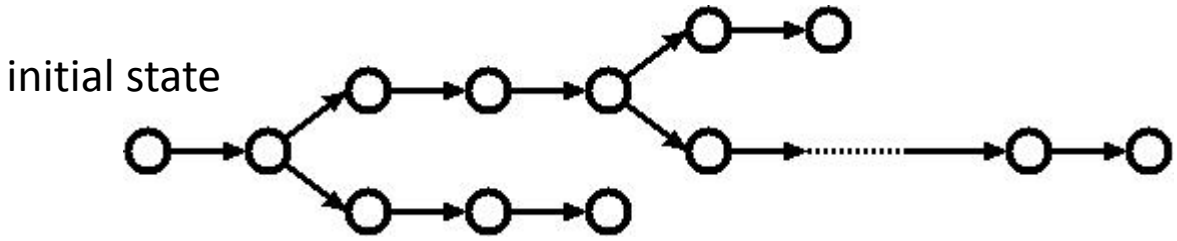
5. Computational Complexity

5.3. Class NP

5.3.*. Nondeterministic computation

- From the viewpoint of the computation tree of a Turing Machine:
  - Computation tree of a deterministic Turing machine forms a path;

  - Computation tree of a *nondeterministic* Turing machine forms a *tree*;



    - each computation halts in an accept/reject state or loop.
    - it accepts if the tree has at least one “accept” in poly-length.

5.計算量の理論

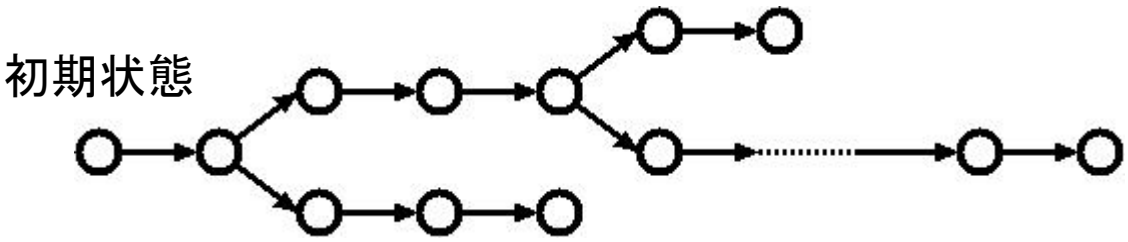
5.3.クラスNP

5.3.*.非決定性計算とは

- チューリングマシンの計算木の観点からみると:

- 決定性のチューリングマシンの計算木はパス(一本道);

- 非決定性のチューリングマシンの計算木は木;



- 各計算プロセスは受理/拒否状態になるか無限ループ
- 木が多項式長の範囲で受理状態を一つでももてば受理.

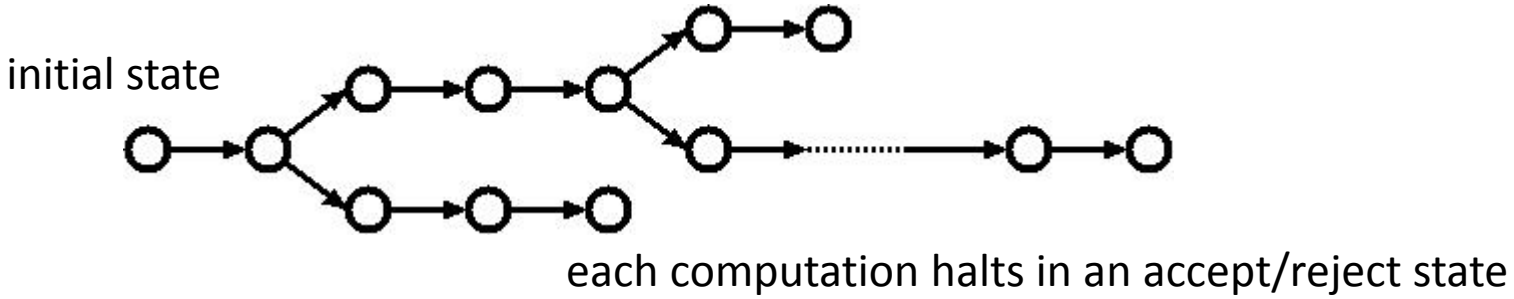
5. Computational Complexity

5.3. Class NP

5.3.*. Nondeterministic computation

The *witness w* gives the right choices

- From the viewpoint of the computation tree of a Turing Machine:
  - Computation tree of a *nondeterministic* Turing machine forms a *tree*;



An NP problem L is recognized by a nondeterministic Turing machine in polynomial time. That is, there is a computation path to an accept state of length polynomial of *n*.

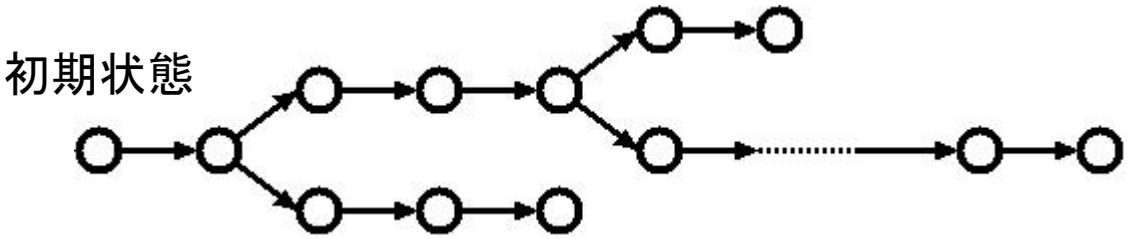
5.計算量の理論

5.3.クラスNP

5.3.*.非決定性計算とは

証拠wは正しい選択肢の列を与える

- チューリングマシンの計算木の観点からみると:
  - 非決定性のチューリングマシンの計算木は木;



- 各計算プロセスは受理/拒否状態になるか無限ループ
- 木が多項式長の範囲で受理状態を一つでももてば受理.

NP 問題 L とは非決定性チューリング機械で
多項式時間で認識できる言語. つまり, 受理状態に至る
n の多項式長の計算パスが存在すればよい.

5. Computational Complexity

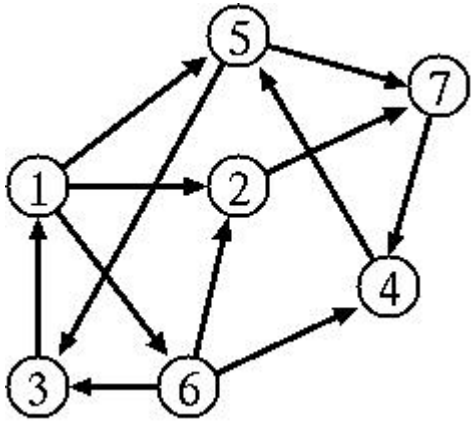
5.3. Class NP

5.3.1. Representative NP problems

- Hamiltonian cycle problem (DHAM)

  Input : <*G*>: a directed graph *G*

  Question : Does *G* have a Hamiltonian cycle?



- We can certainly check all possible permutations of *n*, that counts up to $n! \sim n^n$…
  it takes exponential time.
- If *G* has a Hamiltonian cycle *C*, and we have it as a witness, we can check that it surely a Hamiltonian cycle.

5.計算量の理論

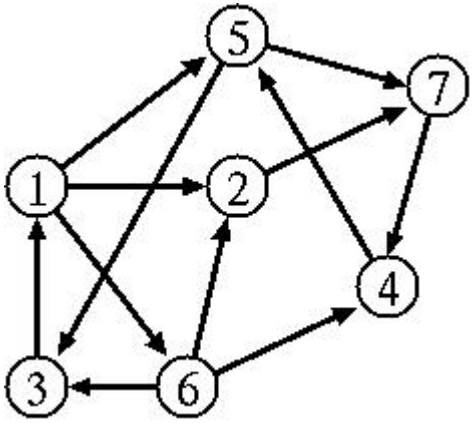
5.3.クラスNP

5.3.1. 代表的なNP問題

- 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?



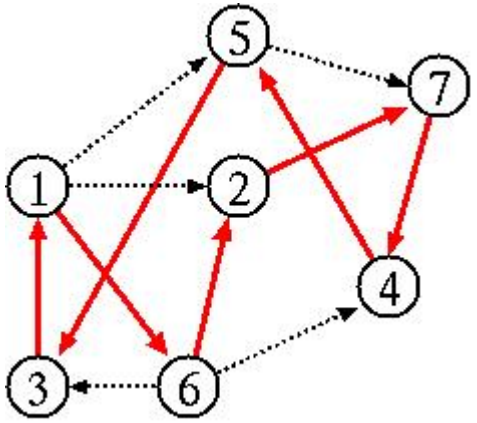
- 原理的には*n*の順列をすべて試せばよいが，可能な組合せの数は最大で$n!$～n^n…
  指数時間かかってしまう.
- もし *G* がハミルトン閉路Cをもつならこれを証拠にすれば，効率よくそれをチェックすることができる.

Note: The Hamiltonian Problem can be solved in $O(n^2 2^n)$ time by using smart algorithm based on "Dynamic Programming" technique

1,6,2,7,4,5,3,1

5.3. Class NP

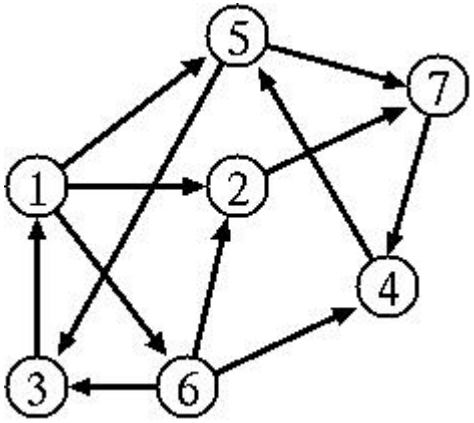
5.3.1. Representative NP problems



- Hamiltonian cycle problem (DHAM)

  Input : <*G*>: a directed graph *G*

  Question : Does *G* have a Hamiltonian cycle?



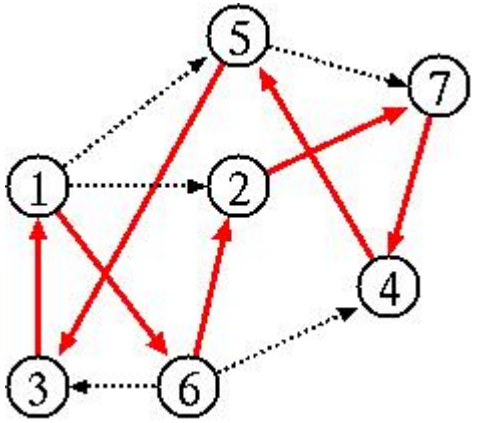
- We can certainly check all possible permutations of *n*, that counts up to $n! \sim n^n$…
  it takes exponential time.
- If *G* has a Hamiltonian cycle *C*, and we have it as a witness, we can check that it surely a Hamiltonian cycle.

1,6,2,7,4,5,3,1

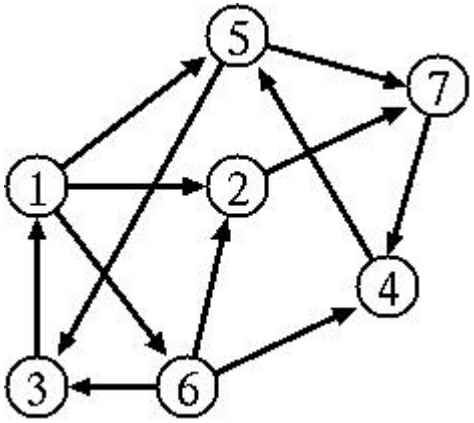
5.計算量の理論

5.3.クラスNP

5.3.1.代表的なNP問題



- 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?



- 原理的には*n*の順列をすべて試せばよいが，可能な組合せの数は最大で$n!$～n^n…
  指数時間かかってしまう.
- もし *G* がハミルトン閉路Cをもつならこれを証拠にすれば，効率よくそれをチェックすることができる.

5. Computational Complexity

5.3. Class NP

5.3.1. Representative NP problems

- SAT, *k*SAT, ExSAT (Satisfiability)

  Input : <*F*> *F* is conjunctive normal form

  Question : Any assignment s. t. $F(a_1, a_2, \ldots, a_n) = 1$?

- If *F* is satisfiable by an assignment *A*, and we have it as a witness, we can check it in polynomial time by the same way as the PROP_EVAL.

- We can certainly check all possible assignments of $(a_1, a_2, \ldots, a_n)$. The assignments are 2^n, that takes exponential time.

5.計算量の理論

5.3.クラスNP

5.3.1.代表的なNP問題

- SAT, *k*SAT, ExSAT (充足可能性)

  入力 : <*F*> *F* は和積標準形命題論理式

  質問 : $F(a_1, a_2, \ldots, a_n) = 1$となる割当ては存在?

- *F* を充足する割当て *A* があるなら,
  それを証拠として使い, PROP_EVALのときと同じ方法で
  多項式時間でチェックできる.

- もちろん$(a_1, a_2, \ldots, a_n)$のすべての可能な割当てを
  チェックすることはできるが, 可能な割当ての個数は
  2^n なので, 指数時間かかる.

5. Computational Complexity

5.3. Class NP

5.3.2. Another aspect of the NP problems

- What does it mean by being an NP set?
  - Using q and R satisfying the predicate characterizing an 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 $q(|x|)$, we can accept or reject them.
Here note that there are $2^{q(|x|)}$ (exponentially many) such strings.

We may think that those sets recognizable as above are NP sets.

5.計算量の理論

5.3.クラスNP

5.3.2. NP問題を別の視点から見る

- NP 集合であることの意味は?
  - •命題述語論理によるNP 集合の特徴付けで出てきた q と R を使うと,「$x \in L$?」という質問に次のアルゴリズムで答えることができる.

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

長さ高々$q(|x|)$のすべての文字列を辞書式に列挙してチェックすれば, 受理または拒否を判断できる.
ただし, こうした文字列は$2^{q(|x|)}$ (指数関数的) 通りある.
こうしたアルゴリズムで認識できる集合をNP集合と考えてもよい.

5. Computational Complexity

5.3. Class NP

5.3.3. More representative NP problems

- Knapsack Problem (KNAP)

  Input : n+1 tuple of natural numbers $< a_1, a_2, \dots, a_n, b>$

  Question: Is there a set of indices $S \subseteq \{1, \dots, n\}$ s.t. $\sum_{i \in S} a_i = b$?

- Bin Packing Problem (BIN)

  Input : n+2 tuple of natural numbers $< a_1, a_2, \dots, a_n, b, k>$

  Question: Is there a partition of a set of indices $U=\{1, \dots, n\}$ into $U_1, \dots, U_k$ such that $\sum_{i \in U_j} a_i \leq b$ 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.計算量の理論

5.3. クラスNP

5.3.3. 代表的なNP問題再び

- **ナップサック問題 (KNAP)**

  入力: 自然数のn+1個組$< a_1, a_2, \dots, a_n, b >$

  質問: 添え字の集合$S \subseteq \{1, \dots, n\}$で$\sum_{i \in S} a_i = b$ を満たすものはあるか?

- **ビン詰め問題 (BIN)**

  入力: 自然数のn+2個組$< a_1, a_2, \dots, a_n, b, k >$

  質問: 添え字の集合 $U=\{1, \dots, n\}$ の分割$U_1, \dots, U_k$で $\sum_{i \in U_j} a_i \leq b$ を満たすものはあるか?

- **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上に大きさ k の頂点被覆は存在するか?

> 頂点被覆 Sとは，各辺$\{u,v\}$に対してu,vの少なくともどちらか一方をふくむ頂点集合

5. Computational Complexity

5.4. Class coNP

5min. Test: Explain why P=coP?

Definition

A set *L* is in coNP if and only if its complement belongs to NP.

Theorem

For every set *L*, the following conditions are equivalent.

(a) $L \in$ coNP

(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*.

[Note] It is nonsense to define coP since it is equal to P.

5.計算量の理論

5.4. クラスcoNP

定義

集合 L が coNP に属する必要十分条件は,
その補集合がNPに属すること.

定理

任意の集合 L に対して, 以下の二つは同値である.

(a) $L \in$ coNP

(b) Lは多項式qと多項式時間で計算できる述語Qを使って
次のように書ける: $L = \{x : \forall w \in \Sigma^* : \ |w| \leq q(|x|)[Q(x,w)]\}$

[注意] coP はPと同値であることがすぐにわかるので,
定義しても無意味.

5. Computational Complexity

5.5. Relations in the Complexity Classes

Theorem $P \subseteq E \subseteq EXP$

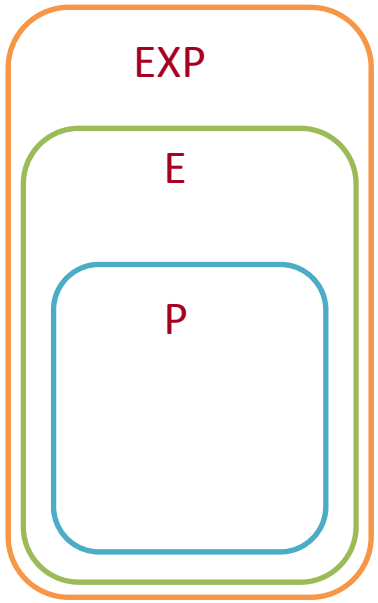
Proof: Obvious from the definition.

Theorem $P \subsetneq E \subsetneq EXP$

Proof: Out of scope in this class…

(Brief idea: We can use *diagonalization* to show a hierarchy theorem that says $TIME(t_1(n)) \subsetneq TIME(t_2(n))$ for, e.g., $t_1(n)^3 = O(t_2(n))$).

We have a *proper* hierarchy



5.計算量の理論

5.5.計算量クラスの関係

定理 $P \subseteq E \subseteq EXP$

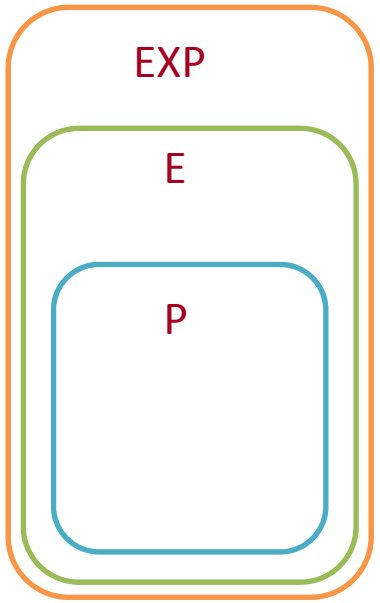
証明: 定義より明らか.

定理 $P \subsetneq E \subsetneq EXP$

証明: 本講義の範囲を超えるので省略.
(アイデアの概略: 対角線論法を巧妙に
使うと, 例えば$t_1(n)^3=O(t_2(n))$といった関数に
対して次の階層定理を示すことができる.

$\mathrm{TIME}(t_1(n)) \subsetneq \mathrm{TIME}(t_2(n))$

*真に異なる*階層構造
が成立する



5. Computational Complexity

5.5. Relations in the Complexity Classes

Theorem

(1) $P \subseteq NP$, $P \subseteq coNP$ ($\therefore P \subseteq NP \cap coNP$)

(2) $NP \subseteq EXP$, $coNP \subseteq EXP$ ($\therefore NP \cup coNP \subseteq EXP$)

Proof (Outline):

(1) $P \subseteq NP$ ($P \subseteq coNP$ is similar)

Ignoring the "witness" in the definition of NP, we immediately obtain the definition of P.

(2) $NP \subseteq EXP$ ($coNP \subseteq EXP$ is similar)

For the "witness" *w* of length *m*, we can check all possible strings of length *m* in exponential time.

5.計算量の理論

5.5.計算量クラスの関係

定理
(1)P ⊆ NP, P ⊆coNP (∴P ⊆ NP ∩ coNP)
(2)NP ⊆ EXP, coNP ⊆ EXP (∴ NP ∪ coNP ⊆ EXP)

証明(概略):
(1) P⊆NP (P⊆coNP も同様)
NPの定義の中の「証拠」を無視すれば,
Pの定義と同値なものが得られる.
(2) NP⊆EXP (coNP⊆EXP も同様)
長さmのすべての文字列に対して
それが長さmの「証拠」wになるかどうかを
指数時間かけてチェックすればよい.

5. Computational Complexity

5.5. Relations in the Complexity Classes

Theorem

(1) NP ⊆ coNP → NP = coNP

(2) coNP ⊆ NP → NP = coNP

(3) NP ≠ coNP → P ≠ NP

Note: From (3), proof for NP ≠ co-NP is harder than that for P ≠ NP.

Proof :

(1) NP ⊆ coNP → NP = coNP

By assumption, it is sufficient to show that coNP ⊆ NP.

We will prove $L \in \text{NP}$ for any $L \in \text{coNP}$.

$L \in \text{coNP} \Leftrightarrow \overline{L} \in \text{NP}$ (by Definition)

$\rightarrow \overline{L} \in \text{coNP}$ (NP ⊆ co-NP)

$\Leftrightarrow L \in \text{NP}$ (Definition and $L=\overline{\overline{L}}$)

5.計算量の理論

5.5.計算量クラスの関係

定理
(1) NP ⊆ coNP → NP = coNP
(2) coNP ⊆ NP → NP = coNP
(3) NP ≠ coNP → P ≠ NP

注: (3)よりNP ≠ co-NP の証明はP ≠ NPの証明よりも難しい.

証明 :

(1) NP ⊆ coNP → NP = coNP

仮定より coNP ⊆ NPを示せばよい.

そこで任意の $L \in$ coNPに対して $L \in$ NP を示す.

$L \in$ coNP $\Leftrightarrow \overline{L} \in$ NP (定義より)

$\rightarrow \overline{L} \in$ coNP (NP ⊆ co-NP)

$\Leftrightarrow L \in$ NP (定義と $L=\overline{\overline{L}}$ より)

5. Computational Complexity

5.5. Relations in the Complexity Classes

Theorem

(1) NP $\subseteq$ coNP $\rightarrow$ NP = coNP

(2) coNP $\subseteq$ NP $\rightarrow$ NP = coNP

(3) NP $\neq$ coNP $\rightarrow$ P $\neq$ NP

Note: From (3), proof for NP $\neq$ co-NP is harder than that for P $\neq$ NP.

Proof: (3) NP $\neq$ coNP $\rightarrow$ P $\neq$ NP

Contraposition : P = NP $\rightarrow$ NP = coNP

If we assume P=NP, for any L we have

$L \in \text{NP} \Leftrightarrow L \in \text{P}$ (P = NP)

$\Leftrightarrow \bar{L} \in \text{P}$ (P = coP)

$\Leftrightarrow \bar{L} \in \text{NP}$ (P = NP)

$\Leftrightarrow L\ (=\bar{\bar{L}}) \in \text{coNP}$ (Definitions of NP/coNP)

$\therefore$ NP = coNP Q.E.D.

5.計算量の理論

5.5.計算量クラスの関係

定理

(1) NP ⊆ coNP → NP = coNP

(2) coNP ⊆ NP → NP = coNP

(3) NP ≠ coNP → P ≠ NP

注: (3)よりNP ≠ co-NP の証明はP ≠ NPの証明よりも難しい.

証明: (3) NP ≠ coNP → P ≠ NP

以下の対偶を示す: P = NP → NP = coNP

P=NPと仮定すると, 任意の集合 L に対して以下を得る

$L \in$ NP $\Leftrightarrow$ $L \in$ P (P = NP)

$\Leftrightarrow$ $\bar{L} \in$ P (P = coP)

$\Leftrightarrow$ $\bar{L} \in$ NP (P = NP)

$\Leftrightarrow$ L $(=\bar{\bar{L}}) \in$ coNP (NP/coNPの定義より)

∴ NP = coNP Q.E.D.

5. Computational Complexity

5.5. Relations in the Complexity Classes

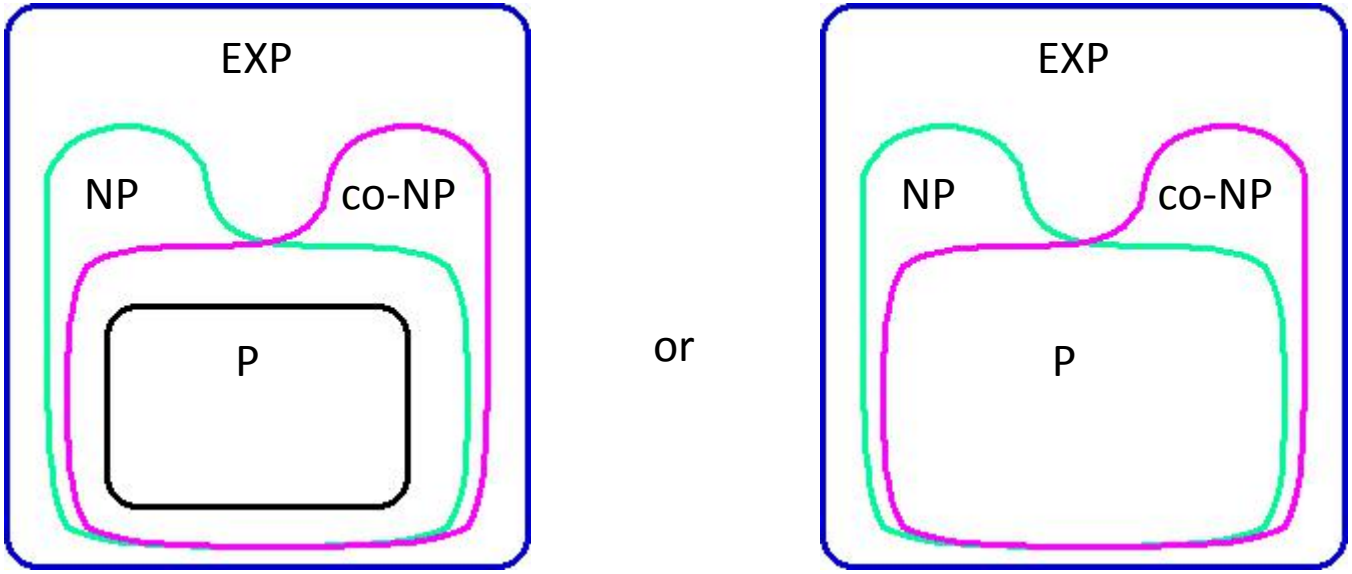
Theorem

(1) NP ⊆ coNP → NP = coNP

(2) coNP ⊆ NP → NP = coNP

(3) NP ≠ coNP → P ≠ NP

We strongly believe that P ≠ NP, and then we have

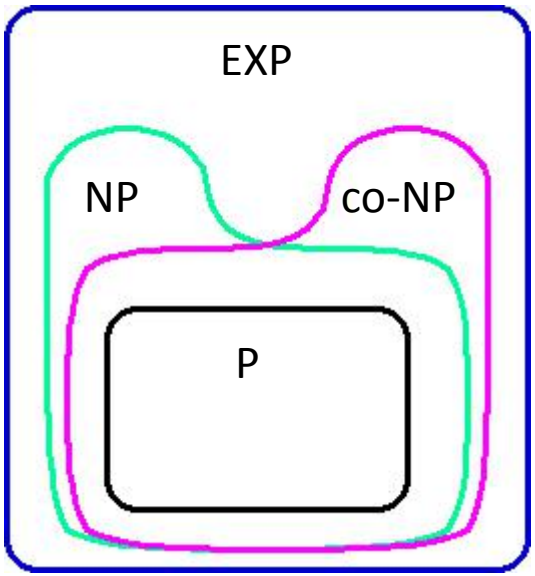


5.計算量の理論

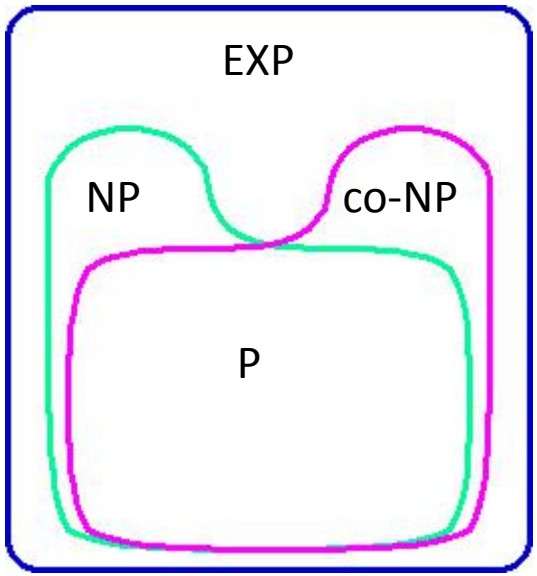
5.5. 計算量クラスの関係

定理
(1) NP ⊆ coNP → NP = coNP
(2) coNP ⊆ NP → NP = coNP
(3) NP ≠ coNP → P ≠ NP

P≠ NPが成立すると強く信じられているので,
以下の構造になっていると予想される.



または



5. Computational Complexity

- Observation of the classes

Definition: Class P
Set L is in the class P ⇔
There exists a poly-time computable predicate R such that
for each $x \in \Sigma^*$, $x \in L \Leftrightarrow R(x)$

Definition: Class NP
Set L is in the class NP ⇔
There exists a poly q and a poly-time computable predicate R such that
for each $x \in \Sigma^*$, $x \in L \Leftrightarrow \exists w \in \Sigma^* : |w| \leqq q(|x|)[R(x,w)]$

Definition: Class coNP
Set L is in the class coNP ⇔
There exists a poly q and a poly-time computable predicate R such that
for each $x \in \Sigma^*$, $x \in L \Leftrightarrow \forall w \in \Sigma^* : |w| \leqq q(|x|)[R(x,w)]$

5.計算量の理論

- **計算量クラスの定義を概観すると…**

クラスPの定義
集合LがクラスPに入る ⇔
以下を満たす多項式時間計算可能述語Rが存在:
各 $x \in \Sigma^*$で$x \in L \Leftrightarrow R(x)$

クラスNPの定義
集合LがクラスNPに入る ⇔
以下を満たす多項式qと多項式時間計算可能述語Rが存在:
各 $x \in \Sigma^*$で$x \in L \Leftrightarrow \exists w \in \Sigma^* : |w| \leqq q(|x|)[R(x,w)]$

クラスcoNPの定義
集合LがクラスcoNPに入る ⇔
以下を満たす多項式qと多項式時間計算可能述語Rが存在:
各 $x \in \Sigma^*$で$x \in L \Leftrightarrow \forall w \in \Sigma^* : |w| \leqq q(|x|)[R(x,w)]$

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Definition

Let *A* and *B* be arbitrary sets.

(1) function h: $A \rightarrow B$: polynomial-time reduction

$\Leftrightarrow$
- (a) h is a total function from Σ^* onto Σ^*
- (b) $x \in \Sigma^* [x \in A \leftrightarrow h(x) \in B]$
- (c) h is polynomial-time computable.

(2) When there is a poly-time reduction from *A* to *B*, we say *A* is polynomial-time reducible to *B*.
Then, we denote by $A \leq_m^P B$

(…within polynomial time, hardness of *A* $\leqq$ that of *B*)

6. 多項式時間計算可能性の解析手法

6.1. 多項式時間還元可能性

定義

A と B を任意の集合とする.

(1) 関数 $h: A \to B$ が多項式時間還元である

$\Leftrightarrow$
- (a) h は Σ^* から Σ^* への全域関数である
- (b) $x \in \Sigma^* [x \in A \leftrightarrow h(x) \in B]$
- (c) h は多項式時間計算可能である.

(2) A から B への多項式時間還元が存在するとき

A は B へ多項式時間還元可能であるといい,

$A \leq_m^P B$ とかく.

(…多項式時間程度の差を無視すれば，A の難しさ≦ B の難しさ)

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Theorem $A \leq_m^P B$ leads to

(1) $B \in$ P → $A \in$ P.
(2) $B \in$ NP → $A \in$ NP.
(3) $B \in$ co-NP → $A \in$ coNP.
(4) $B \in$ EXP → $A \in$ EXP.

Note : class E is exceptional. Generally, $B \in$ E → $A \in$ E is not true.

Ex.: When we define ONE≡{1}, for each set L in P we have

$$L \leq_m^P \mathrm{ONE}$$

if we define $h(x) \equiv \begin{cases} 1, & \text{if } x \in L \\ 0, & \text{otherwise} \end{cases}$

6.多項式時間計算可能性の解析手法

6.1.多項式時間還元可能性

定理　$A \leq_m^P B$ のとき次が成立する

(1) $B \in$ P → $A \in$ P.
(2) $B \in$ NP → $A \in$ NP.
(3) $B \in$ co-NP → $A \in$ coNP.
(4) $B \in$ EXP → $A \in$ EXP.

注意：クラス E は例外．一般に $B \in$ E → $A \in$ E は成立しない．

例: ONE≡{1} と定義すると，Pの各集合Lに対して，

$$L \leq_m^P \mathrm{ONE}$$

である．ここで $h(x) \equiv \begin{cases} 1, & \text{if } x \in L \\ 0, & \text{otherwise} \end{cases}$

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Theorem *A, B, C*: arbitrary sets

(1) $A \leq_m^P A$

(2) $A \leq_m^P B \wedge B \leq_m^P C \rightarrow A \leq_m^P C$

Definition

$$A \equiv_m^P B \leftrightarrow A \leq_m^P B \wedge B \leq_m^P A$$

$\equiv_m^P$ is an equivalence relation.

6.多項式時間計算可能性の解析手法

6.1.多項式時間還元可能性

定理 *A, B, C*を任意の集合とする.

(1) $A \leq_m^P A$

(2) $A \leq_m^P B \wedge B \leq_m^P C \rightarrow A \leq_m^P C$

定義

$A \equiv_m^P B \leftrightarrow A \leq_m^P B \wedge B \leq_m^P A$

$\equiv_m^P$ は同値関係.

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Theorem

(1) $2\mathrm{SAT} \leq_m^P 3\mathrm{SAT} \leq_m^P \mathrm{SAT} \leq_m^P \mathrm{ExSAT}$

(2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

Proof

(1) we have some proofs depending on definition:

(a) each instance of 2SAT is also in 3SAT if the definition is “at most 3 literals in a clause”.

(b) each clause $(x \vee y)$ can be replaced by $(x \vee y \vee y)$.

(c) each clause $(x \vee y)$ can be replaced by $(x \vee y \vee z) \wedge (x \vee y \vee \bar{z})$.

In any case, they are poly-time reduction, and the original formula is satisfiable iff so is the resulting formula.

6.多項式時間計算可能性の解析手法

6.1.多項式時間還元可能性

定理

(1) $2\mathrm{SAT} \leq_m^P 3\mathrm{SAT} \leq_m^P \mathrm{SAT} \leq_m^P \mathrm{ExSAT}$

(2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

証明

(1) 定義によっていくつかの証明が考えられる:

(a) 定義が「各項に高々3リテラル」の場合は，2SATの入力は3SATの入力としても有効なので，特に示すことはない.

(b) 各項 $(x \vee y)$ を単に $(x \vee y \vee y)$ で置き換えればよい.

(c) 各項 $(x \vee y)$ に対して新しい変数を導入して

$(x \vee y \vee z) \wedge (x \vee y \vee \bar{z})$.

と置き換えてもよい.

どの場合も多項式時間還元で，元の論理式が充足可能である必要十分条件は，新しい式が充足可能であること.

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Theorem

(1) $\mathrm{2SAT} \leq_m^P \mathrm{3SAT} \leq_m^P \mathrm{SAT} \leq_m^P \mathrm{ExSAT}$

(2) $\mathrm{3SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

Proof (Outline)

(2) It is sufficient to show that $\mathrm{ExSAT} \leq_m^P \mathrm{3SAT}$ by (1).

Strategy:

For any given *F* in ExSAT, we construct another *F'* in 3SAT such that *F* is satisfiable iff *F'* is satisfiable.

To do that, we first construct the computation tree of *F*, and construct *F'* that represents the computation process of *F*.

6.多項式時間計算可能性の解析手法

6.1.多項式時間還元可能性

定理

(1) $\mathrm{2SAT} \leq_m^P \mathrm{3SAT} \leq_m^P \mathrm{SAT} \leq_m^P \mathrm{ExSAT}$

(2) $\mathrm{3SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

証明 (概略)

(2) (1)より，$\mathrm{ExSAT} \leq_m^P \mathrm{3SAT}$ が成立することを示せばよい.

基本戦略:

ExSATの式 *F* が与えられたら，それに基づいて3SATの式*F'* を構成する. ただしここで*F*が充足可能である必要十分条件が*F'*が充足可能であるようにする. そのために，まず*F*の計算木を構築し，次に*F*の計算手順を表現する論理式*F'*を構築する.

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

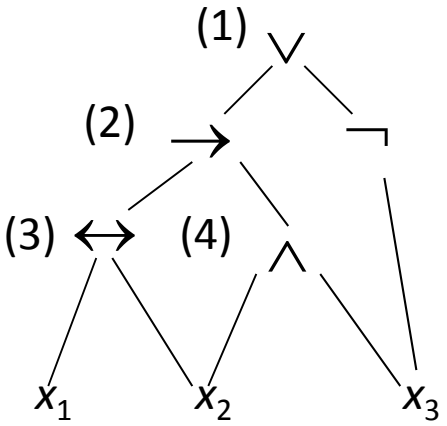
Theorem (2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

Proof (Outline)

(2) It is sufficient to show that $\mathrm{ExSAT} \leq_m^P 3\mathrm{SAT}$ by (1).

Reduction from ExSAT to 3SAT by an example:

$$F(x_1, x_2, x_3) \equiv [[x_1 \leftrightarrow x_2] \rightarrow [x_2 \wedge x_3]] \vee \neg x_3$$



$$(1) V_1 \equiv V_2 \vee \neg x_3$$

$$(2) V_2 \equiv [V_3 \rightarrow V_4]$$

$$(3) V_3 \equiv [x_1 \leftrightarrow x_2]$$

$$(4) V_4 \equiv x_2 \wedge x_3$$

6.多項式時間計算可能性の解析手法

6.1.多項式時間還元可能性

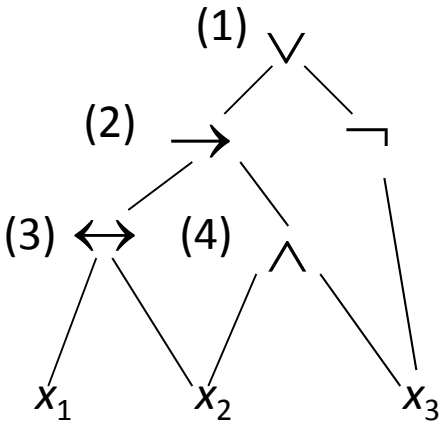
定理 (2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

証明 (概略)

(2) $\mathrm{ExSAT} \leq_m^P 3\mathrm{SAT}$ が成立することを示せばよい.

ExSAT から 3SAT への還元を例で示す:

$$F(x_1, x_2, x_3) \equiv [[x_1 \leftrightarrow x_2] \rightarrow [x_2 \wedge x_3]] \vee \neg x_3$$



$$(1) V_1 \equiv V_2 \vee \neg x_3$$

$$(2) V_2 \equiv [V_3 \rightarrow V_4]$$

$$(3) V_3 \equiv [x_1 \leftrightarrow x_2]$$

$$(4) V_4 \equiv x_2 \wedge x_3$$

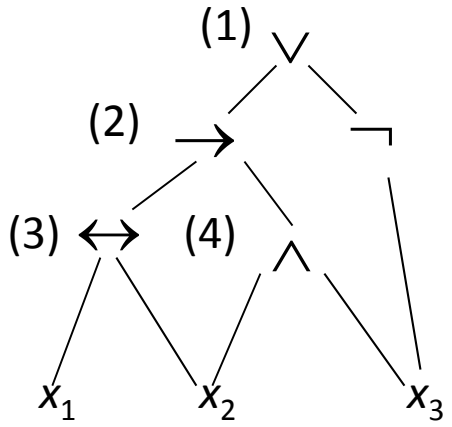
6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Theorem (2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

Reduction from ExSAT to 3SAT by an example:

$$F(x_1, x_2, x_3) \equiv [[x_1 \leftrightarrow x_2] \rightarrow [x_2 \wedge x_3]] \vee \neg x_3$$



$$(1) V_1 \equiv V_2 \vee \neg x_3$$
$$(2) V_2 \equiv [V_3 \rightarrow V_4]$$
$$(3) V_3 \equiv [x_1 \leftrightarrow x_2]$$
$$(4) V_4 \equiv x_2 \wedge x_3$$

$$F''(x_1, x_2, x_3) \equiv U_1 \wedge [U_1 \leftrightarrow [U_2 \vee \neg x_3]] \wedge [U_2 \leftrightarrow [U_3 \rightarrow U_4]]$$
$$\wedge [U_3 \leftrightarrow [x_1 \leftrightarrow x_2]] \wedge [U_4 \leftrightarrow [x_2 \wedge x_3]]$$

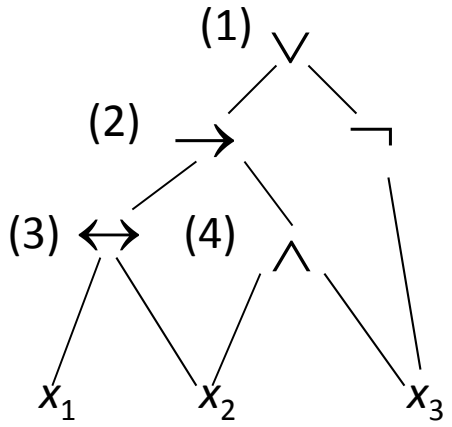
6.多項式時間計算可能性の解析手法

6.1.多項式時間還元可能性

定理 (2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

ExSAT から 3SAT への還元を例で示す:

$$F(x_1, x_2, x_3) \equiv [[x_1 \leftrightarrow x_2] \rightarrow [x_2 \wedge x_3]] \vee \neg x_3$$



$$(1) V_1 \equiv V_2 \vee \neg x_3$$
$$(2) V_2 \equiv [V_3 \rightarrow V_4]$$
$$(3) V_3 \equiv [x_1 \leftrightarrow x_2]$$
$$(4) V_4 \equiv x_2 \wedge x_3$$

$$F''(x_1, x_2, x_3) \equiv U_1 \wedge [U_1 \leftrightarrow [U_2 \vee \neg x_3]] \wedge [U_2 \leftrightarrow [U_3 \rightarrow U_4]]$$
$$\wedge [U_3 \leftrightarrow [x_1 \leftrightarrow x_2]] \wedge [U_4 \leftrightarrow [x_2 \wedge x_3]]$$

6. Analysis on Polynomial-Time Computability

6.1. Polynomial-time Reducibility

Theorem (2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

Reduction from ExSAT to 3SAT by an example:

$$F''(x_1, x_2, x_3) \equiv U_1 \wedge [U_1 \leftrightarrow [U_2 \vee \neg x_3]] \wedge [U_2 \leftrightarrow [U_3 \rightarrow U_4]] \wedge [U_3 \leftrightarrow [x_1 \leftrightarrow x_2]] \wedge [U_4 \leftrightarrow [x_2 \wedge x_3]]$$

Then, by constriction, *F*() is satisfiable iff *F''*() is satisfiable.
We show *F''*() can be represented by an equivalent *F'*() in 3SAT.

$$\begin{aligned} U_1 \leftrightarrow [U_2 \vee \neg x_3] &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee \neg [U_2 \vee \neg x_2]] \\ &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee [\neg U_2 \wedge x_2]] \\ &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee \neg U_2] \wedge [U_1 \vee x_2] \\ &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee \neg U_2 \vee \neg U_2] \wedge [U_1 \vee x_2 \vee x_2] \end{aligned}$$

The other cases are similar, and *F'*() is in 3SAT.

6.多項式時間計算可能性の解析手法

6.1. 多項式時間還元可能性

定理 (2) $3\mathrm{SAT} \equiv_m^P \mathrm{SAT} \equiv_m^P \mathrm{ExSAT}$

ExSAT から 3SAT への還元を例で示す:

$$F''(x_1, x_2, x_3) \equiv U_1 \wedge [U_1 \leftrightarrow [U_2 \vee \neg x_3]] \wedge [U_2 \leftrightarrow [U_3 \rightarrow U_4]] \wedge [U_3 \leftrightarrow [x_1 \leftrightarrow x_2]] \wedge [U_4 \leftrightarrow [x_2 \wedge x_3]]$$

このとき構成から，*F*() は充足可能⇔*F''*()は充足可能.
F''() をこれと同値な3SATの要素 *F'*() で表現する.

$$\begin{aligned} U_1 \leftrightarrow [U_2 \vee \neg x_3] &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee \neg [U_2 \vee \neg x_2]] \\ &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee [\neg U_2 \wedge x_2]] \\ &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee \neg U_2] \wedge [U_1 \vee x_2] \\ &= [\neg U_1 \vee U_2 \vee \neg x_3] \wedge [U_1 \vee \neg U_2 \vee \neg U_2] \wedge [U_1 \vee x_2 \vee x_2] \end{aligned}$$

他のケースも同様に変形でき，*F'*() は3SATの要素となる.

6. Analysis on Polynomial-Time Computability

6.2. Completeness

6.2.1. Definition and basic properties

Definition

For a class C, if a set A satisfies

(a) $\forall L \in \mathrm{C}\ [L \leq_m^P A]$,

the set A is called C-hard (under $\leq_m^P$).

Moreover, if we have

(b) $A \in \mathrm{C}$,

then A is called C-complete.

Ex. Examples of NP-complete sets

3SAT, SAT, ExSAT, DHAM, KNAP, BIN, VC, etc.

6.多項式時間計算可能性の解析手法

6.2.完全性

6.2.1. 定義と基本性質

定義

クラスCに対して，集合Aが次を満たすとき

(a) $\forall L \in \mathrm{C}\ [L \leq_m^P A]$

集合 A は($\leq_m^P$のもとで) C困難であるという.

さらに次を満たすなら

(b) $A \in \mathrm{C}$

A はC完全であるという.

例. NP完全集合の例

3SAT, SAT, ExSAT, DHAM, KNAP, BIN, VC など

6. Analysis on Polynomial-Time Computability

6.2. Completeness

6.2.1. Definition and basic properties

Theorem. For any C-hard (or C-complete) set A,

(1) $A \in \mathrm{P} \rightarrow \mathrm{C} \subseteq P$ CP: $\mathrm{C} \not\subset \mathrm{P} \rightarrow A \notin \mathrm{P}$

(2) $A \in \mathrm{NP} \rightarrow \mathrm{C} \subseteq NP$ CP: $\mathrm{C} \not\subset \mathrm{NP} \rightarrow A \notin \mathrm{NP}$

(3) $A \in \mathrm{coNP} \rightarrow \mathrm{C} \subseteq coNP$ CP: $\mathrm{C} \not\subset \mathrm{coNP} \rightarrow A \notin \mathrm{coNP}$

(4) $A \in \mathrm{EXP} \rightarrow \mathrm{C} \subseteq EXP$ CP: $\mathrm{C} \not\subset \mathrm{EXP} \rightarrow A \notin \mathrm{EXP}$

Proof: CP: contraposition

(1) Let B be any C-set. Then, since A is C-hard,

$B \leq_m^P A$ and by the assumption $A \in \mathrm{P}$, we have $B \in \mathrm{P}$

(2), (3), (4) are similar.

6.多項式時間計算可能性の解析手法

6.2.完全性

6.2.1.定義と基本性質

定理　C困難(またはC完全)な任意の集合Aに対して,

(1) $A \in$ P → C $\subseteq P$　　対偶: C$\not\subset$ P → $A \notin$P

(2) $A \in$ NP → C $\subseteq NP$　　対偶: C$\not\subset$NP → $A \notin$NP

(3) $A \in$ coNP → C $\subseteq coNP$　　対偶: C$\not\subset$ coNP → $A \notin$ coNP

(4) $A \in$ EXP → C $\subseteq EXP$　　対偶: C$\not\subset$EXP → $A \notin$EXP

証明:

(1) 任意のC集合をBとする. AがC困難であることから,

$B \leq_m^P A$ であり, $A \in$Pという仮定より $B \in$Pをえる.

(2), (3), (4) も同様.

6. Analysis on Polynomial-Time Computability

6.2. Completeness

6.2.1. Definition and basic properties

Theorem. For any C-hard (or C-complete) set A,

(1) $A \in P \rightarrow C \subseteq P$ CP: $C \not\subset P \rightarrow A \notin P$

(2) $A \in NP \rightarrow C \subseteq NP$ CP: $C \not\subset NP \rightarrow A \notin NP$

(3) $A \in coNP \rightarrow C \subseteq coNP$ CP: $C \not\subset coNP \rightarrow A \notin coNP$

(4) $A \in EXP \rightarrow C \subseteq EXP$ CP: $C \not\subset EXP \rightarrow A \notin EXP$

Ex. : Meaning of Theorem for class NP

Let A be NP-complete set.

By the contraposition of Theorem (1) we have

$NP \neq P \rightarrow A \notin P$

That is, NP-complete sets are NP-sets that cannot be recognized in polynomial time unless P = NP.

6.多項式時間計算可能性の解析手法

6.2.完全性

6.2.1.定義と基本性質

定理　C困難(またはC完全)な任意の集合Aに対して,

(1) $A \in$ P → C $\subseteq P$　　対偶: C$\not\subset$ P → $A \notin$ P

(2) $A \in$ NP → C $\subseteq NP$　　対偶: C$\not\subset$NP → $A \notin$ NP

(3) $A \in$ coNP → C $\subseteq coNP$　対偶: C$\not\subset$ coNP → $A \notin$ coNP

(4) $A \in$ EXP → C $\subseteq EXP$　　対偶: C$\not\subset$EXP → $A \notin$ EXP

例 : クラスNPに関する定理の意味するところ

NP完全集合をAとする.

定理(1)の対偶より:　NP≠P ➔ $A \notin$ P

つまり, NP完全集合はP=NPでない限り,

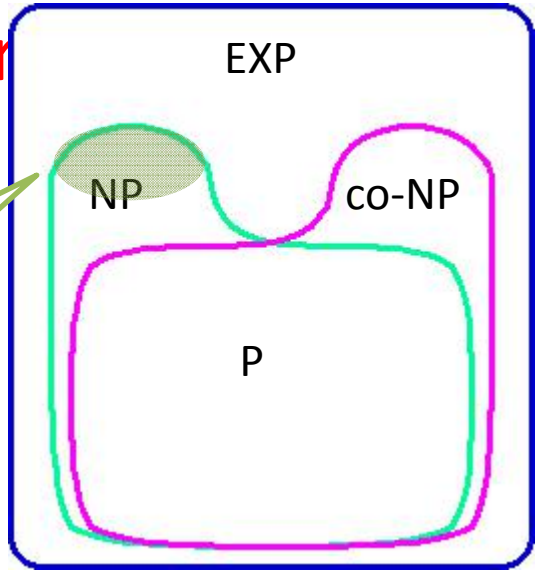
多項式時間では認識できないNP集合である.

6. Analysis on Polynomial-Time Computability

6.2. Completeness

6.2.1. Definition and basic proper

NP-complete problems form the most difficult problems in the class NP.



Ex. : Meaning of Theorem for class NP

Let *A* be NP-complete set.

By the contraposition of Theorem (1) we have

$NP \neq P \rightarrow A \notin P$

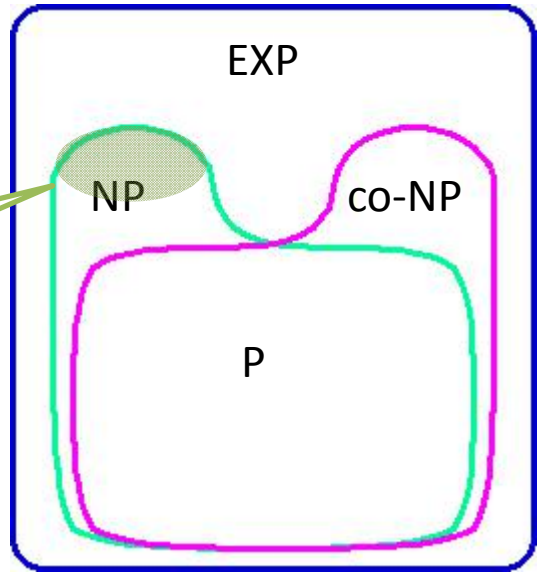
That is, NP-complete sets are NP-sets that cannot be recognized in polynomial time unless P = NP.

6.多項式時間計算可能性の解析手法

6.2.完全性

6.2.1.定義と基本性質

NP完全問題とは，クラスNPの中で最も難しい問題群を構成しているといえる.



例：クラスNPに関する定理の意味するところ

NP完全集合をAとする.

定理(1)の対偶より：　NP≠P ➔ $A \notin$ P

つまり，NP完全集合はP=NPでない限り，

多項式時間では認識できないNP集合である.

6. Analysis on Polynomial-Time Computability

6.2. Completeness

6.2.1. Definition and basic properties

Theorem 6.4. *A*: any C-complete set

For any set *B* we have

(1) $A \leq_m^P B \rightarrow B$ is C-hard.

(2) $A \leq_m^P B$ and $B \in \mathrm{C} \rightarrow B$ is C-complete.

Proof :

By definition, $\forall L \in \mathrm{C}[L \leq_m^P A]$

By Theorem, $L \leq_m^P A \wedge A \leq_m^P B \rightarrow L \leq_m^P B$

Therefore, $\forall L \in \mathrm{C}[L \leq_m^P B]$

That is, *B* is C-hard.

Once you have an NP-complete problem *A*, it can be used to measure to the other problems

6.多項式時間計算可能性の解析手法

6.2. 完全性

6.2.1.定義と基本性質

定理　A:任意のC完全集合

任意の集合 B に対して以下が成立

(1) $A \leq_m^P B$ →B は C困難.

(2) $A \leq_m^P B$ かつ $B \in$ C → B は C完全.

証明:

定義より, $\forall L \in \mathrm{C}[L \leq_m^P A]$

定理より, $L \leq_m^P A \wedge A \leq_m^P B \rightarrow L \leq_m^P B$

よって, $\forall L \in \mathrm{C}[L \leq_m^P B]$

つまりBはC困難.

ひとたびNP完全問題 Aが得られたら, これを使って他の問題の困難性を「測定」できる.