アナウンス（覚書）

- レポート(2)：出題は5月22日，締切は5月29日
- 5月24日のチュートリアルアワーは補講をします
- 期末試験：5月31日チュートリアルアワー
  - 前回と同じ形式？
  - 出題範囲は「計算量」のみにする予定

I238 計算の理論

上原 隆平

2018年I-1期（4-5月）

I238 Computation Theory

by

Prof. Ryuhei Uehara

Term I-1, April-May, 2018

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

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

5.計算量の理論

5.1.計算時間の評価

5.1.3.問題の時間計算量

定義: 自然数上の関数 $t(n)$ に対して，時間計算量 $O(t(n))$ の集合(認識問題)全体の集合を **$O(t(n))$時間計算量クラス** とよび，TIME(t(n)) とかく. こうした関数 $t(n)$ は制限時間と呼ぶ.

5.2.代表的な計算量クラス

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

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

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

C集合： 計算量クラスCに入る集合.
C問題： C集合の認識問題

ある問題がPに入っていないなら、現実的には手に負えない…

5. Computational Complexity

5.1. Time Complexity Classes

5.1.3. Time complexity of a problem

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

5.2. Representative time complexity classes

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

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

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

C set : set in the complexity class C.

C problem : problem of recognizing a C set.

Problems not in P are intractable from the practical viewpoint…

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**<u>共通で自然な</u>**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

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"

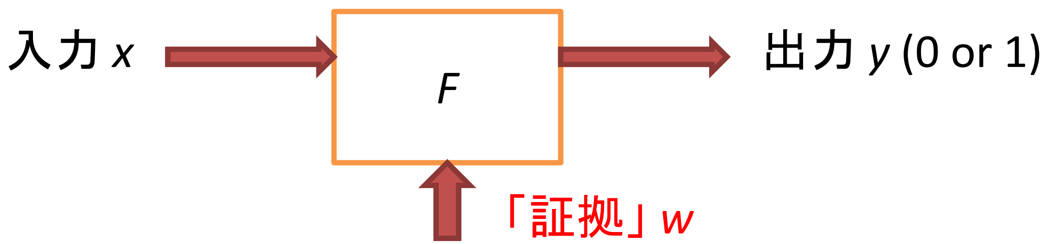
5.計算量の理論

5.3.クラスNP

5.3.*.非決定性計算とは

「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∈*L* を認識する

c.f.: NP=**<u>N</u>**ondeterministic **<u>P</u>**olynomial

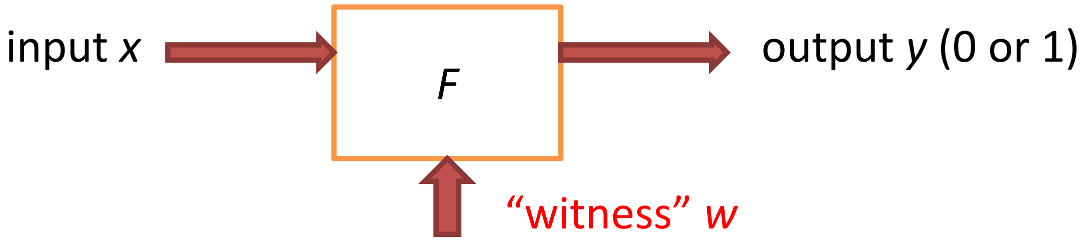
5. Computational Complexity

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」

- 論理の視点からみると:

この文字列 w を「証拠」とよぶ

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：

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

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

“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

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

i.e.,

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

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

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

「非決定性計算」

- チューリング機械の視点から見ると:

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

- 非決定性選択は複数の選択肢のうち，
「どれかが真」ならば真になる．

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

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

5.3.*. Nondeterministic computation

"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 "<u>one of two cases (0) and (1)</u>" 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

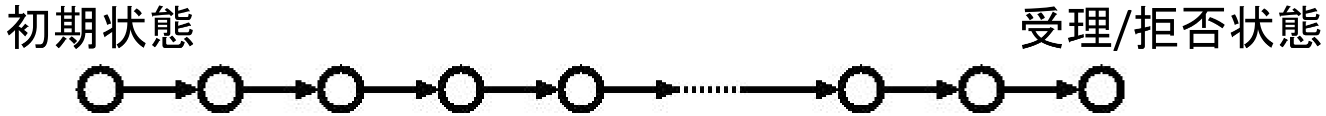
5.計算量の理論

5.3.クラスNP

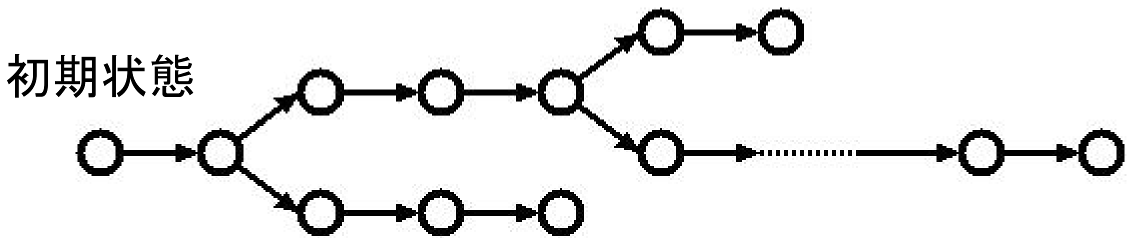
5.3.*.非決定性計算とは

- チューリング機械の計算木の観点からみると:

  - 決定性のチューリング機械の計算木はパス(一本道);



  - 非決定性のチューリング機械の計算木は木;



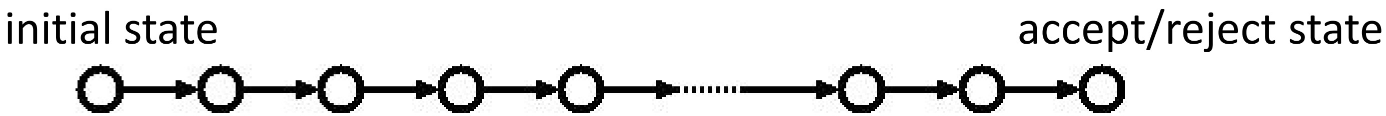
- 各計算プロセスは受理/拒否状態になるか無限ループ
- 木が多項式長の範囲で受理状態を一つでももてば受理.

5. Computational Complexity

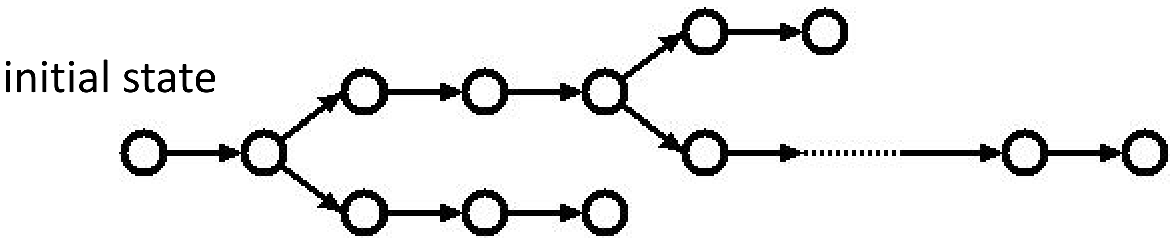
5.3. Class NP

5.3.*. Nondeterministic computation

- 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.

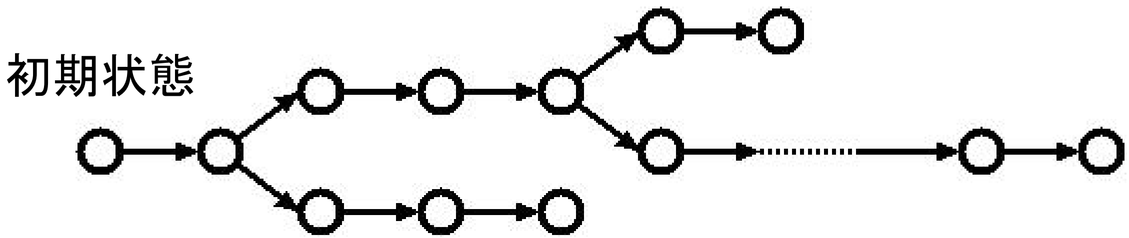
5.計算量の理論

5.3.クラスNP

証拠wは正しい選択肢の列を与える

5.3.*.非決定性計算とは

- チューリング機械の計算木の観点からみると:
  - 非決定性のチューリング機械の計算木は木;



- 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.

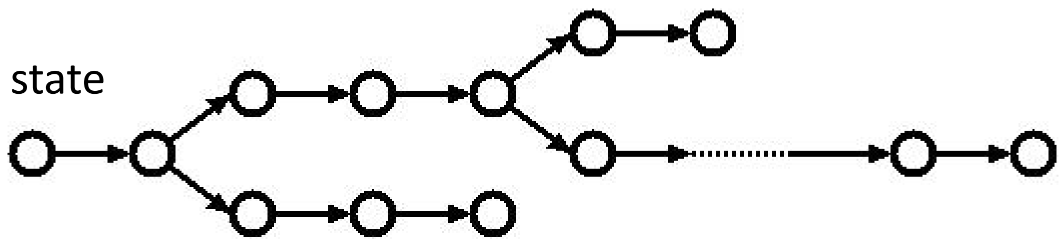
5. Computational Complexity

5.3. Class NP

The *witness* w gives the right choices

5.3.*. Nondeterministic computation

- 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.*.非決定性チューリング機械の形式的定義

決定性チューリング機械の形式的定義（復習）:

チューリング機械とは7つ組$(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$で,

Qは状態集合(有限集合)

Σはブランク記号bを含まない入力アルファベット(有限集合)

Γはテープアルファベットでb∈ΓかつΣ⊆Γ (有限集合)

$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$は遷移関数

$q_0 \in Q$は初期状態

$q_a \in Q$は受理状態

$q_r \in Q$は拒否状態で, $q_a \neq q_r$

5. Computational Complexity

5.3. Class NP

5.3.*. Formal definition of nondeterministic Turing Machine

Formal definition of deterministic Turing machine (again) :

A Turing machine is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, where

- Q is the set of states,
- Σ is the input alphabet not containing the blank b,
- Γ is the tape alphabet, where $b \in \Gamma$ and $\Sigma \subseteq \Gamma$,
- $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$ is the transition function,
- $q_0 \in Q$ is the start state,
- $q_a \in Q$ is the accept state, and
- $q_r \in Q$ is the reject state, where $q_a \neq q_r$

5.計算量の理論

5.3.クラスNP

決定性チューリング機械:

– 状態遷移は:
  - 関数$\delta:Q\times\Gamma\rightarrow Q\times\Gamma\times\{L,R\}$により決まる.

非決定性チューリング機械:

– 状態遷移は:
  - 関数$\delta:Q\times\Gamma\rightarrow 2^{Q\times\Gamma\times\{L,R\}}$により決まる.

ある状態における「次の状態」が複数あってよい.

5. Computational Complexity

5.3. Class NP

Deterministic Turing machine :

A transition is:

- Determined by the function $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$

Nondeterministic Turing machine :

– A transition is :

- Determined by the function $\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$

Many “next states” can exist for each state.

5.計算量の理論

5.3.クラスNP

決定性チューリング機械:

– 計算とは:
  - 初期状態 q_0 からはじまり,
  - 与えられたテープ上の文字列で状態遷移し,
  - q_a か q_r に到達したら停止する.
  - q_a で停止したら入力を受理, q_r で停止したら入力を拒否と考える.

非決定性チューリング機械:

– 計算とは:
  - 初期状態 q_0 からはじまり,
  - 与えられたテープ上の文字列で状態の集合に遷移し,
  - 遷移した状態集合が q_a を含んでいたら受理して停止する.
    – 上記以外の場合は定義する必要がない(モデルによって色々だが, どれも同じ)

5. Computational Complexity

5.3. Class NP

Deterministic Turing machine :

– A deterministic computation:
  - It starts from the start configuration q_0,
  - makes transitions on a given tape string, and
  - halts when it reach to q_a or q_r.
  - It accepts on q_a, or rejects on q_r.

Nondeterministic Turing machine :

– A nondeterministic computation :
  - It starts from the start configuration q_0,
  - makes transitions on a given tape string, and
  - halts when it reach to the set of states that contains q_a or q_r.
  - It accepts on q_a, or rejects on q_r.
    – It is not essential the case that the set contains both

5.計算量の理論

5.3.クラスNP

クラスNPの非決定性チューリング機械による定義:

- 集合LがクラスNPに入る必要十分条件は，ある非決定性チューリング機械 M と多項式 q が存在し，以下を満たすとき:

$$x \in L \Leftrightarrow M \text{ が高々 } q(|x|) \text{ 時間で } x \text{ を受理する}$$

5. Computational Complexity

5.3. Class NP

Definition of the class NP by using nondeterministic Turing machine:

- A set L is in the class NP if and only if there are a nondeterministic Turing machine M and a polynomial q such that:

$$x \in L \Leftrightarrow M \text{ accepts } x \text{ at most } q(|x|) \text{ time.}$$

5.計算量の理論

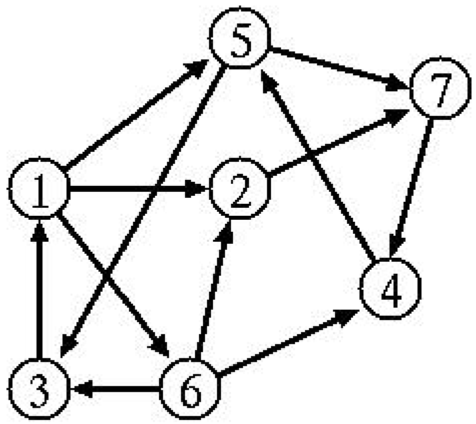
5.3.クラスNP

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

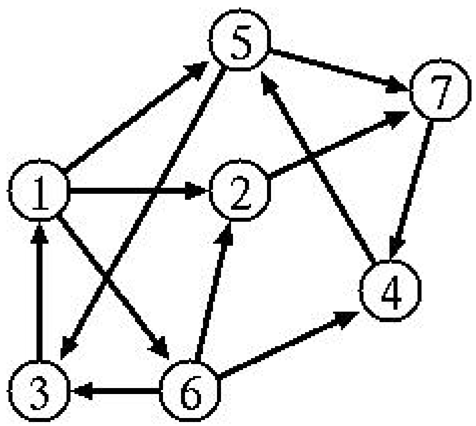
5.3. Class NP

5.3.1. Representative NP problems

- Hamiltonian cycle problem (DHAM)

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

  Question : Does *G* have a Hamiltonian cycle?



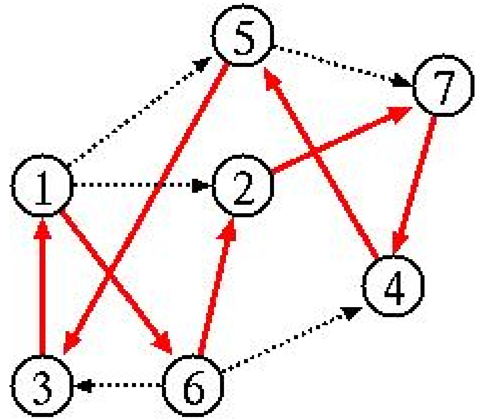
- We can certainly check all possible permutations of n, that counts up to $n! \sim n^n$…
  it takes exponential time.
- If *G* has a Hamiltonian cycle *C*, and we have it as a witness, we can check that it surely a Hamiltonian cycle.

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

5.計算量の理論

5.3.クラスNP

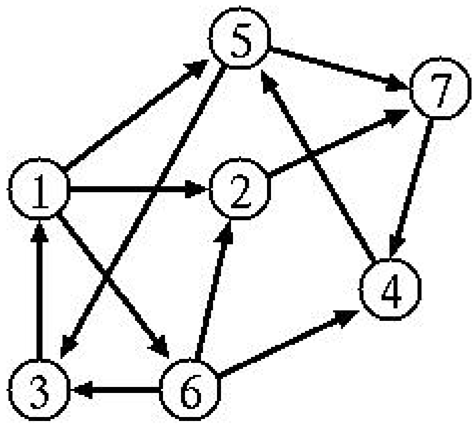
5.3.1.代表的なNP問題



- 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?

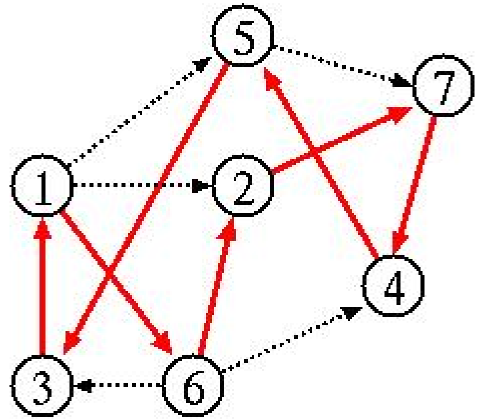


- 原理的には*n*の順列をすべて試せばよいが，可能な組合せの数は最大で$n!\sim n^n$...
 指数時間かかってしまう.
- もし *G* がハミルトン閉路Cをもつならこれを証拠にすれば，効率よくそれをチェックすることができる.

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

5. Computational Complexity

5.3. Class NP

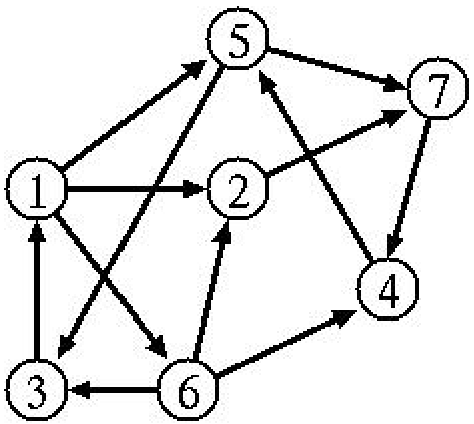


5.3.1. Representative NP problems

- 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

- 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

5.3.1. 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

5.3.2. 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 $w \in \Sigma^{\leq q(|x|)}$ 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

5.3.2. 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 $w \in \Sigma^{\le q(|x|)}$ 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

5.3.3. 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, … , 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, … , a_n, b, k>
  質問: 添え字の集合 $U=\{1, \dots, n\}$ の分割U_1, … , U_kで $\sum_{i \in U_j} a_i \le b$ 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

5.3. Class NP

5.3.3. More representative NP problems

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

[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

5.4. Class coNP

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

5.5.計算量クラスの関係

定理 P $\subseteq$ E $\subseteq$ EXP

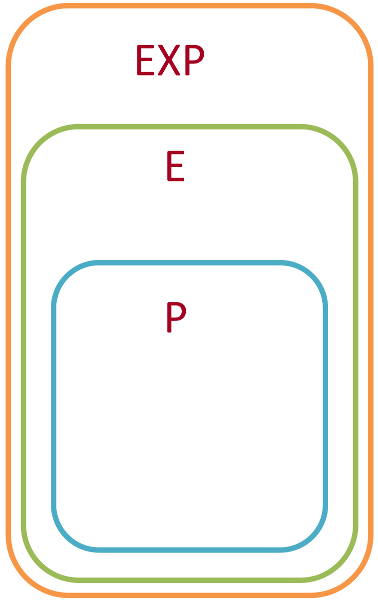
証明: 定義より明らか.

定理 P $\subsetneq$ E $\subsetneq$ EXP

証明: 本書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 $P \subseteq E \subseteq EXP$

Proof: Obvious from the definition.

Theorem $P \subsetneqq E \subsetneqq EXP$

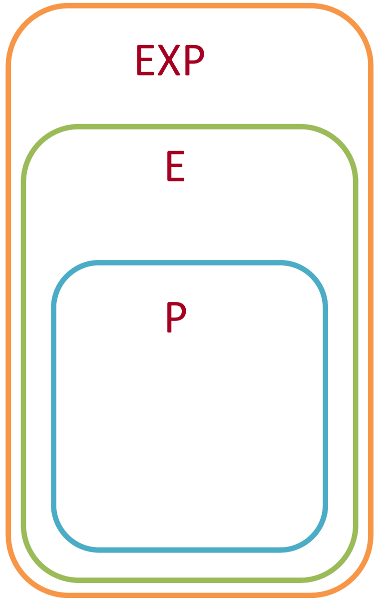
Proof: Out of scope in this class...

(Brief idea: We can use *diagonalization* to show a hierarchy theorem that says

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

for, e.g., $(t_1(\mathrm{n}))^3 = O(t_2(n))$).

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

(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) NP $\subseteq$ coNP → NP = coNP

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

仮定より coNP $\subseteq$ 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 $\subseteq$ 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 ≠ coNP $\rightarrow$ P ≠ NP

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

Proof :

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

By assumption, it is sufficient to show that coNP $\subseteq$ NP.

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

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

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

$\Leftrightarrow L \in$ 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 $\subseteq$ coNP $\rightarrow$ NP = coNP

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

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

以下の対偶を示す: P = NP $\rightarrow$ 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

$$
\begin{aligned}
L \in \mathrm{NP} &\Leftrightarrow L \in \mathrm{P} \quad (\mathrm{P} = \mathrm{NP}) \\
&\Leftrightarrow \bar{L} \in \mathrm{P} \quad (\mathrm{P} = \mathrm{coP}) \\
&\Leftrightarrow \bar{L} \in \mathrm{NP} \quad (\mathrm{P} = \mathrm{NP}) \\
&\Leftrightarrow L\ (=\bar{\bar{L}}) \in \mathrm{coNP} \quad (\text{NP/coNPの定義より})
\end{aligned}
$$

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

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 ≠ coNP $\rightarrow$ P ≠ NP

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

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

$\boxed{\text{Contraposition: } P = NP \rightarrow NP = coNP}$

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

$$
\begin{aligned}
L \in NP \quad &\Leftrightarrow \quad L \in P \quad (P = NP) \\
&\Leftrightarrow \quad \bar{L} \in P \quad (P = coP) \\
&\Leftrightarrow \quad \bar{L} \in NP \quad (P = NP) \\
&\Leftrightarrow \quad L\ (= \bar{\bar{L}}) \in coNP \quad (\text{Definitions of NP/coNP})
\end{aligned}
$$

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

5.計算量の理論

5.5. 計算量クラスの関係

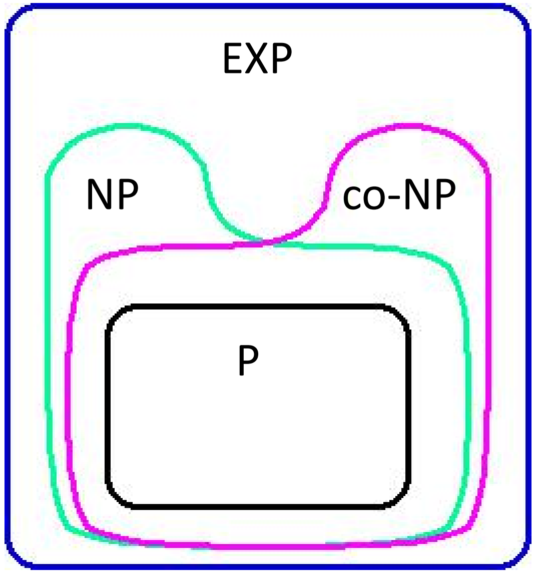
定理

(1) NP ⊆ coNP → NP = coNP

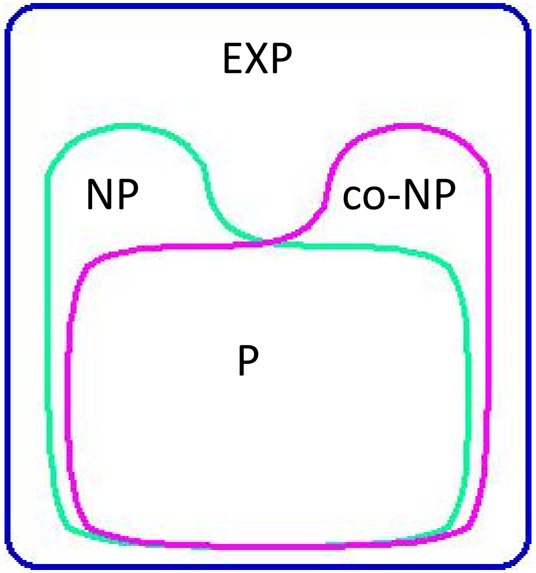
(2) coNP ⊆ NP → NP = coNP

(3) NP ≠ coNP → P ≠ NP

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

5.5. Relations in the Complexity Classes

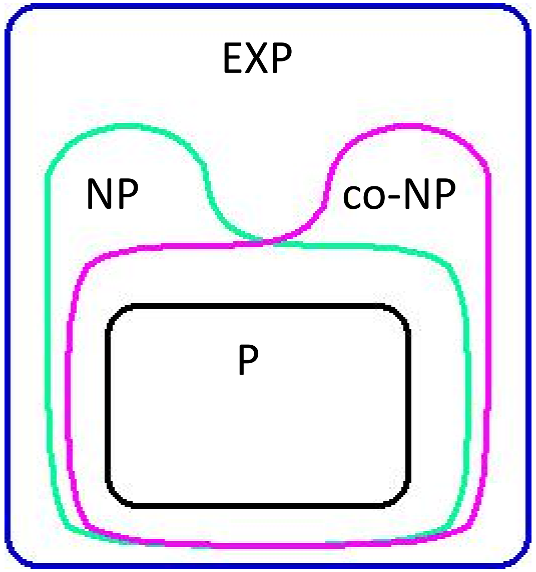
Theorem

(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



or