計算可能性理論（第 II 部 計算複雑性）

改訂版 **2024.03.01**

東京都立大学 理 数理　　鈴木 登志雄

目 次

6　計算複雑性の基本事項

計算可能か不可能かという観点ではなく，計算可能な関数に対して計算の手間がどれぐらいかかるのかを論じよう．

6.1 計算時間の単位

まず，何をもって計算の１ステップとみなすかを決める．このやり方にはいろいろある．

例．

1. 整数の四則計算１回を１ステップとみなす．数論アルゴリズムや暗号理論において，おおらかに計算量を見積もるときにこう考えることがある．

2. 計算モデルとして多テープチューリング計算機を採用し，自然数は２進表記するものとする．そして，チューリング計算機の命令１回の実行（ヘッドが１マス動くことにほぼ等しい）を１ステップとみなす．

3. レジスター マシンといって，CPU を単純化した計算モデルを考え，その基本命令１回の実行（レジスタの値を書き換えてプログラムカウンタの値を１回書き換えることにほぼ等しい）を１ステップとみなす．

4. 標準形プログラムというものを定める．標準形プログラムは while ループを一つもち，その中にプログラムカウンタ pc の値で場合分けする case 文をひとつもつ．Case 文を構成する各行は，簡単な条件分岐の下でプログラムカウンタの値を更新するか，簡単な代入文の後にプログラムカウンタの値を更新するかいずれかのパターンになっている．自然数は２進表記するものとし，データとしてはビット列のみを扱う．標準形プログラムのループ１回分の実行を１ステップとみなす．

ここで１ステップをどう定義するかによって個々の関数の計算量は少しずつ変わってくる．しかし，後で定義する決定性多項式時間計算量クラス P，および非決定性多項式時間計算量クラス NP の概念は，ここでの約束を少々変えても変わらない．文献 [渡辺 1992] にしたがい，ここでは上記 (4) で１ステップを定義する．

定義 **6.1** [渡辺 1992]

1. 標準形プログラム A と入力 x が与えられたときの計算時間とは，A が x を受け取ってから停まるまでのステップ数とする．停まらない場合は無限大とする．

2. 標準形プログラム A の時間計算量は，入力 x のサイズ（２進数としての桁数）に対する関数として定める．ただし，サイズ ℓ に対して，そのサイズの入力のうち計算時間を最大とするものに注目して，その計算時間を採用する．A が k 入力であり（k 個の入力をもち）$k \geq 2$ である場合，入力 $x_1, \cdots, x_k$ のサイズはこれらのサイズの和であると定義する．

注　(1) ２進表記された自然数 x の桁数は，$\log x$（底は２）にほぼ等しい．(2) １０進表記における桁数と２進表記における桁数は定数倍の違いでしかない．(3) 有向グラフを入力として考えるとき，たとえば隣接行列などを用いて，グラフをビット列でコード化する．

時間計算量を論じるとき，定数倍やプラス定数の違いにこだわっても報われないことが多い．それらは，計算モデルとして何を採用するか，入力のコード化としてどんな方法を採用するかによって変わってしまうからである．そこで，オーダー記法を導入する．

定義 **6.2** [渡辺 1992] f, g は $\mathbb{N}$ から $\mathbb{N}$ への関数であるとする．

$$\exists c, d > 0 \ \forall n \ f(n) \leq cg(n) + d \tag{6.1}$$

であるとき，「f はオーダー g である」，あるいは「f は $O(g)$ である」という．

注　(1) $O(g)$ を関数族とみなして，「f は $O(g)$ である」とは「$f \in O(g)$」の意味であるととるのが妥当に思える．しかし，慣習として「$f = O(g)$」と表記する．

(2) $f = O(g)$ を定義する式 (6.1) において「$\forall n$」を「有限個を除いてすべての n に対して（記号では $\forall^{\infty} n$）」で置き換えても同値な定義となる．たとえばすべての $n \geq 10$ に対して $f(n) \leq cg(n)+d$ となるとき，$d' = \max\{d, f(0), \ldots, f(9)\}$ とおけばすべての自然数 n に対して $f(n) \leq cg(n) + d'$ となる．

(3) ある N 以上のすべての n に対して $g(n) \neq 0$ であるとき，「$f = O(g)$」の必要十分条件は「（$n \geq N$ の範囲で）f/g が有界」であることである．

6.2　自然数に対する四則計算の時間計算量

自然数に対する簡単な計算の時間計算量を見積もろう．

1. 加法 $m+n$ は（自然数 m，n の関数として．以下同様）$O(\log m + \log n)$ 時間計算可能である．
2. 乗法 $m \cdot n$ は $O(\log m \cdot \log n)$ 時間計算可能である．
3. 整数の除法に関して商を求める演算 $m \ \mathrm{div} \ n$ は $O(\log m \cdot \log n)$ 時間計算可能である．
4. 整数の除法に関して剰余を求める演算 $m \bmod n$ は $O(\log m \cdot \log n)$ 時間計算可能である．

7　計算量クラス

7.1　基本的な計算量クラス

計算量理論ではビット列を基本的な対象と考えることが多い．ビット列を語（word）ともいう．単に集合と言えばビット列の集合を指すことが多い．いま考えている集合がビット列の集合であることを強調したいとき，その集合を言語（language） あるいはオラクル（oracle）とよぶことがある．一方，言語の集合をクラス（class）とよぶことが多い．「クラス」という言葉の使い方が集合論とは違うので注意してほしい．

定義 **7.1**　1. プログラム A が言語 L を決定するとは，A が L の特性関数を計算することをいう．L の特性関数は $\{0,1\}^*$ を定義域とする関数であり，$x \in L$ のとき $f(x)=1$，そうでないとき $f(x)=0$ となるものである．

2. （計算量クラス **DTIME** [AB2009, p.25]）T は $\mathbb{N}$ から $\mathbb{N}$ への関数であるとする．$\mathbf{DTIME}(T(n))$ とは，以下の条件を満たす言語 L 全体のクラスであるとする：あるプログラム A が存在して，A は L を決定し，かつ，A の時間計算量は $O(T(n))$ である．

3. （多項式時間計算量クラス **P** [AB2009, p.25]）ここで p は非負整数係数の多項式全体に渡って動くものとする．

$$\mathbf{P} = \bigcup_{p} \mathbf{DTIME}(p(n))$$

言語 L が **P** に属するとき，L は（決定性）多項式時間計算可能であるという．

4. （2のリニア乗時間計算量クラス **E** [渡辺 1992, p.128]）

$$\mathbf{E} = \bigcup_{c \geq 1} \mathbf{DTIME}(2^{cn})$$

5. （指数時間計算量クラス **EXP** [渡辺 1992, p.128]）ここで p は非負整数係数の多項式全体に渡って動くものとする．

$$\mathbf{EXP} = \bigcup_{p} \mathbf{DTIME}(2^{p(n)})$$

注 (1) $\mathbf{P}$ は以下のクラスと一致する．

$$\bigcup_{c\geq 1}\mathbf{DTIME}(n^c)$$

証明の概要：p は非負整数係数の多項式であるとする．p の次数を k とすると $p(x)/x^k$ は有界である．

(2) 同様にして，$\mathbf{EXP}$ は以下のクラスと一致する．

$$\bigcup_{c\geq 1}\mathbf{DTIME}(2^{n^c})$$

(3) $\mathbf{P}\subseteq\mathbf{E}$ が成り立つ．

証明の概要：$c,d\geq 1$ とする．ロピタル (l'Hospital) の定理により $x^c/2^{dx}\to 0(x\to\infty)$ である．したがって $x^c/2^{dn}$ は有界である．

(4) $\mathbf{E}\subseteq\mathbf{EXP}$ が成り立つ．

(5) 上記の $\mathbf{DTIME}(T(n))$ を，[渡辺 1992, p.110] では $\mathrm{TIME}(T)$ と書き，$O(T)$ 時間計算量クラスとよんでいる．また，T を制限時間とよんでいる．

(6) 厳密にいうと，[AB2009, p.25, Def. 1.12] では標準形プログラムではなくチューリングマシンを使って $\mathbf{DTIME}(T(n))$ を定義している．これは上記の定義と異なる．しかし，多項式時間計算量クラス $\mathbf{P}$ は同じものになる．

問 **7.1** 有限ビット列全体の集合を $\{0,1\}^*$ で表す．$\{0,1\}^*$ の部分集合 L_1, L_2 は多項式時間計算量クラス P に属するとする．

(1) L_1 の補集合が P に属することを示せ．

(2) $L_1\cap L_2$ が P に属することを示せ．

(3) $L_1\cup L_2$ が P に属することを示せ．

7.2 階層定理

制限時間 T_1，T_2 において T_2 が T_1 より十分大きければ $\mathbf{DTIME}(T_1(n))\subsetneq\mathbf{DTIME}(T_2(n))$ となる．具体的には次の結果が知られている．

定理 **7.2** （階層定理 [渡辺 1992, p.115]）制限時間 T_1，T_2 は任意の正の数 c に対して以下をみたすとする．

$$\exists n_0\forall n\geq n_0\ \ cT_1(n)^2\leq T_2(n)$$

このとき $\mathbf{DTIME}(T_1(n))\subsetneq\mathbf{DTIME}(T_2(n))$ が成り立つ．

上記を用いると以下の関係を示せる．

$$\mathbf{P}\subsetneq\mathbf{E}\subsetneq\mathbf{EXP}$$

8　計算量クラス P に属する集合の例

グラフの頂点を論理式（や，計算機システムの時点表示）とみなす考え方の威力を味わってもらいたい．

8.1　グラフ

グラフは，適当なビット列に翻訳して扱う [渡辺 1992, p.27]．グラフ G の（データとしての）サイズとは，G を表すビット列の長さのことをいう．翻訳のしかたにもよるが，G の頂点の個数を n とするとき，G のサイズは高々 n^2 のオーダーであると考えてよい．本節で扱うグラフは以下の条件を満たすものに限ることにする「どの頂点 s においても s から s 自身への辺はない」．もちろん，s から s への道はあってもよい．ここで言っているのは，s を出てただちに s 自身に戻る辺がないということである．

例 **8.1** 無向グラフの到達可能性問題 ST-CON [渡辺 1992, p.132] は多項式時間計算可能である．

証明の概略 [Pa1994, §1.1] 入力として無向グラフ G および，その頂点 s，t が与えられたとする．s から t へ到達可能かどうかを以下のように判定する．

(1) 初期状態において，G のすべての頂点は黒いとする．まず，頂点 s を黄色に塗る．（黄色は「現在注目している頂点」）

(2) もし黄色の頂点が一つ以上あれば，以下を行う．なければ (3) に行く．

(2a) 黄色の頂点から辺ひとつで到達できる黒い頂点をすべて緑に塗る．（緑は「次に注目する予定の頂点」）

(2b) 黄色の頂点をすべて赤く塗る．（赤は「検査済みの頂点」）

(2c) 緑の頂点をすべて黄色に塗る．そして (2) の冒頭へ戻る．

(3) 頂点 t が黒のままなら「到達不能」と出力して終了し，そうでなければ「到達可能」と出力して終了．

G の頂点の個数を n とする．手順 (1)，(2a)，(2b)，(2c)，(3) の各々を実行する手間が（G のサイズの）多項式で抑えられることは容易にわかる．また，上記の手順 (2) を１回行うごとに赤い頂点は少なくとも一つ増えるから，手順 (2) は最大でも n 回しか行われない．以上により，この手順の時間計算量は（G のサイズの）多項式で抑えられることがわかる．また，s から到達可能な頂点はいずれも，手順 (2) を終了した時点で黒でない色に塗られている．よって上記のアルゴリズムは無向グラフの到達可能性問題を解く．　□

同様にして，有向グラフの到達可能性問題も多項式時間計算可能であることがわかる．

定理 **8.1** G は連結な有向グラフとする．このとき，以下の二つの条件は同値である．

(I) G はオイラー閉路をもつ．（オイラー閉路とは，すべての辺をちょうど一度通って出発点に戻る道）．

(II) G のどの頂点においても，その頂点から出る矢印の数（outdegree）と，その頂点へ入る矢印の数（indegree）が一致する．

証明の概略 [リュー 1986, §5.6]（あるいは，適当な離散数学の入門書を参照） (I)→(II) の証明は容易．

(II)→(I) の証明：辺（矢印）の個数に関する帰納法で示す．辺が２本以下のときは容易．次に辺が k 本（$k > 2$）の場合を考える．(II) を仮定する．最初は G のすべての辺が黒いとする．また，G の頂点には重複なく番号が振ってあるものとする．以下の手順に従って前処理を行う．

(1) まず，もっとも番号の若い頂点 s を「現在の頂点」とする．

(2) もし「現在の頂点」から出る黒い矢印が一つ以上あれば，以下を行う．なければ終了する．

(2a) 上記のような矢印のうち，その終点（s でない方の頂点）の番号がもっとも若いものを赤く塗る．

(2b) 「現在の頂点」を s ではなく，いま塗った矢印の終点に更新する．そして (2) の冒頭へ戻る．

G は条件 (II) をみたすから，前処理が終わった時点の「現在の頂点」は s（最初の頂点）でなければならない．したがって，この時点ですべての辺が赤ければ，前処理で塗った道がオイラー閉路である．そうでないときは，黒い辺だけで構成される部分グラフ（複数の連結成分に分かれている場合は，その各々）に帰納法の仮定を適用する．すると赤い部分グラフについてのオイラー閉路と，黒い部分グラフ（たち）についてのオイラー閉路が得られる．両者を結合すると G のオイラー閉路になる． □

例 **8.2** 連結な有向グラフの一筆描き問題 DEULER [渡辺 1992, p.132] は多項式時間計算可能である．

証明の概略 与えられたグラフのすべての頂点について outdegree と indegree が一致するかどうか調べる．一致しない頂点が一つでもあればその時点で調査をやめ「オイラー閉路なし」と出力して終了する．そうでなければ（すべての頂点において outdegree と indegree が一致すれば）「オイラー閉路あり」と出力して終了する．この手順の時間計算量は（G のサイズの）多項式で抑えられる．また定理 8.1 により，上記手順が有向グラフの一筆描き問題を解くことがわかる． □

8.2 命題論理式

適当な文字コードを定めることにより，命題論理式もビット列に翻訳して扱う．命題論理式 F のサイズとは，F を表すビット列の長さのことをいう．翻訳のしかたにもよるが，F に現れる記号（命題変数，論理記号，括弧）の総数を n とするとき，F のサイズは高々 $n \log_2 n$ のオーダーであると考えてよい．

定義 **8.2** [渡辺 1992, p.12] (1) 命題変数，またはそれに否定記号（$\neg$）を付けたものをリテラル（**literal**）という．

(2) リテラル有限個を「または」の記号（$\vee$）でつないだ式を節（**clause**）という．和項ともいう．

(3) 節有限個を「かつ」の記号（$\wedge$）でつないだ式を連言標準形（**conjunctive normal form**）という．和積式，**CNF** ともいう．

(4) k を自然数とする．CNF のうち，どの節もちょうど k 個のリテラルからなるものを kCNF（k 和積式）という．（節の個数はいくつでもよい．）

例 **8.3** (1) リテラルの例 X_1

(2) リテラルの例 $\neg X_2$

(3) 節の例 $X_1 \vee \neg X_2 \vee X_3$

(4) 2CNF の例 $(X_1 \vee \neg X_2) \wedge (\neg X_3 \vee X_2) \wedge (\neg X_4 \vee \neg X_1)$

(5) 3CNF の例 $(X_1 \vee \neg X_2 \vee X_3) \wedge (\neg X_3 \vee X_2 \vee \neg X_4)$

定義 **8.3** [渡辺 1992, p.132] (1) 各々の命題変数に対して真（1），偽（0）の値を与えること（すなわち，注目している命題変数全体の集合から $\{1, 0\}$ への関数）を真理値割り当て（**truth assignment**）という．あるいは，より簡潔に「割り当て」といったり，「付値」ともいう．

(2) 命題論理式 F に対して，それを真にする割り当てが少なくとも一つ存在するとき，F は充足可能（**satisfiable**）であるという．充足可能な式全体の集合を SAT で表す．

(3) k を自然数とする．kCNF であって，なおかつ充足可能でもある式全体の集合を kSAT で表す．また，kCNF であるような式を入力として与え，「それが充足可能であるか？」を問う問題も kSAT で表す．

例 **8.4** (1) 充足可能な式の例 $X_1 \vee X_2$

(2) 充足不可能な式の例 $X_1 \wedge \neg X_1$

(3) 2SAT に属する式の例 $(\neg X_1 \vee X_2) \wedge (\neg X_2 \vee X_1) \wedge (X_1 \vee X_3)$

(4) 2CNF であるが 2SAT には属さない式の例 $(\neg X_1 \vee X_2) \wedge (\neg X_2 \vee \neg X_1) \wedge (X_1 \vee X_3) \wedge (\neg X_3 \vee X_1)$

問 **8.1** 例 8.4 (3) の式（それを F_3 とよぼう）が充足可能であること，および (4) の式（F_4）が充足不能であることを，真理値表を書いて確かめよ．

与えられた命題論理式 F が充足可能かどうかは真理値表を書くことによってわかる．しかし，この方法では時間計算量が指数オーダーになってしまう．なぜなら，F に現れる命題変数を $X_1, \cdots, X_n$ とするとき，これらに真（1），偽（0）を割り当てる方法は 2^n 個あるからである．にもかかわらず，以下の定理が成立する．

定理 **8.4** $2\mathrm{SAT}$ は多項式時間計算可能である．

証明の概略 [Pa1994, §9.2] 以下では，論理式の中に二重否定が現れたとき，その都度，二重否定を除去して簡約化するものとする．たとえば，リテラル $\neg X_1$ に否定記号を付けたときは，$\neg\neg X_1$ の二重否定を除去して X_1 にする．さて，2CNF であるような式 F が入力として与えられたとしよう．ここで F に現れる命題変数，およびその否定形のすべてからなる集合を V とする．V を頂点集合とする有向グラフ G を以下のように定める．$\ell_1, \ell_2 \in V$ に対し，頂点 ℓ_1 から ℓ_2 へ向かう辺（矢印）があるための必要十分条件は，F の節で $\ell_1 \to \ell_2$ と同値なもの（すなわち，$(\neg\ell_1) \vee \ell_2$）が存在することとする．

ここで，以下の二つの条件 (I)，(II) が同値であることを示そう．

(I) G の少なくとも一つの頂点 x に対して，x から $\neg x$ に向かう道と $\neg x$ から x に向かう道がともに存在する．

(II) F は充足不能．

(I)→(II) の証明：(I) を仮定する．W を真理値割り当てとする．まず，W において x が真の場合を考える．x から $\neg x$ に至る道 $\Gamma = \ell_0\ell_1\cdots\ell_m$ に着目する（$\ell_0 = x$ かつ $\ell_m = \neg x$）．Γ の始点 x は W において真（1）であり，終点 $\neg x$ は W において偽（0）である．そこで，「W において ℓ_{i+1} が偽」となる最小の番号を i とする．このとき W において ℓ_i は真である．したがって，W において $\ell_i \to \ell_{i+1}$ は偽である．ところが F の節の一つは $\ell_i \to \ell_{i+1}$ と同値なのだから，W において F は偽である．W において x が偽の場合は，$\neg x$ から x に至る道に対して上記と同様に議論する．するとこの場合もやはり W において F は偽である．以上により，いかなる真理値割り当て W に対しても F は偽である．言い換えると，F は充足不能である．

(II)→(I) の証明：対偶を示す．(I) の否定を仮定する．すなわち，(I) の条件をみたす頂点 x は存在しないとする．割り当てを以下のように構成する．

(1) 初期状態では，V に属するどのリテラルも真理値を与えられていない．

(2) V に属するリテラル ℓ のうち，まだ真理値を与えられておらず，なおかつ ℓ から $\neg\ell$ への道が存在しないようなものがあれば，それをひとつとって以下を行う．複数あるときは，命題変数の添え字がもっとも若いものを最優先し，否定記号がない方を優先して選ぶ．なければ終了する．

(2a) ℓ に真を割り当てる．さらに，ℓ から ℓ' への道があるような頂点 ℓ' で，値が未定のものすべてに真を割り当てる．

(2b) $\neg\ell$ に偽を割り当てる．さらに，ℓ'' から $\neg\ell$ への道があるような頂点 ℓ'' で，値が未定のものすべてに偽を割り当てる．

(i) 上記の手順により，F に現れる命題変数すべてに対して真理値（真または偽の値）が割り当てられる．

(i) の証明：背理法の仮定として真理値を持たない変数 x が残ったとし，それらのうち最も添字の小さい x に注目する．仮定により x は条件 (I) をみたさないから，x から $\neg x$ への道がないか，$\neg x$ から x への道がないかのどちらかが成り立つ．すると x か $\neg x$ のどちらかはステップ (2) の ℓ として選ばれ，x は真理値を与えられる．これは背理法の仮定に反する．以上により背理法で (i) が示された．

したがって上記手順により真理値割り当てが得られる．それを W' としよう。

(ii) W' において F は真になる．

(ii) の証明：背理法の仮定として W' において F が偽とする．このとき F のある節が偽になる。その節はある $\ell \to \ell'$ と同値になる．これが偽なので，ℓ は真，ℓ' は偽である．場合１：ℓ' よりも先に，ℓ に真理値が割り当てられた場合．このとき，(2a) により，ℓ' にも真が割り当てられる．これは ℓ' に偽が割り当てられたことに反する．場合２：ℓ よりも先に，ℓ' に真理値が割り当てられた場合．このとき，(2b) により，ℓ にも偽が割り当てられる．これは ℓ に真が割り当てられたことに反する．以上により背理法で (ii) が示された．

ここまでをまとめると，(i) と (ii) により F は充足可能である．つまり (II) の否定が示された．

以上により，(I) と (II) は同値である．そこで与えられた論理式 F に対し，グラフ G を構成して条件（I）が成り立つかどうか調べることによって，F が充足可能かどうか判定できる．G の構成が（F のサイズの）多項式時間でできることは容易にわかる．例 8.1 と同様にして有向グラフの到達可能性問題が多項式時間計算可能であることがわかる．したがって条件 (I) の判定は多項式時間でできる．よって 2SAT は多項式時間計算可能である． □

問 **8.2** 例 8.4 (3)，(4) の式各々に対し定理 8.4 のグラフ G を描き，条件 (a) が成り立つか確かめよ．

8.3 似ているが難しい問題

ハミルトン閉路問題および，3SAT について多項式時間アルゴリズムは知られていない（ハミルトン閉路とは，すべての頂点をちょうど一度通って出発点に戻る道）．そのようなアルゴリズムは存在しないという予想が優勢である．これらの問題の性質を捉えるために計算量クラス **NP** という概念，および **NP** 完全性という概念が導入されている．

9 非決定性多項式時間計算量クラス NP

有向ハミルトングラフ全体を DHAM と表そう．3SAT や DHAM の計算コストに関する性質をうまくとらえるために計算量クラス **NP** というものが考えられている．本節の多項式は非負整数係数の多項式を意味するとする．

9.1 NP の定義

例 **9.1** 3SAT は計算量クラス **EXP** に属する．以下のアルゴリズムを考えればよい．与えられた命題論理式（のコード）F に対して，F への真理値割り当てをしらみつぶしに生成して F が真となるか調べる．一度でも F が真になればその時点で 1（Yes）を出力して終了する．一度も F が真とならないまま探索が完了した場合は 0（No）を出力して終了する．

正当性：このアルゴリズムが 3SAT を決定することは容易にわかる．

計算量：真理値割り当てのコードの長さは F のコードの長さ $\ell := |F|$ 以下である．したがって真理値割り当ての個数は 2^ℓ 以下である．また，与えられた真理値割り当て u の下で F が真であるかどうかは ℓ の多項式時間で判定できる．よって上記アルゴリズムの時間計算量は $2^\ell \times (\ell$ の多項式$)$ で抑えられる．これは 2 の多項式乗で抑えられる．

例 **9.2** DHAM は計算量クラス **EXP** に属する．以下のアルゴリズムを考えればよい．与えられた有向グラフ（のコード）G に対して，G の頂点の順列をしらみつぶしに生成してハミルトン閉路になっているか調べる．一つでもハミルトン閉路がみつかればその時点で 1（Yes）を出力して終了する．一度もみつからないまま探索が完了した場合は 0（No）を出力して終了する．

正当性：このアルゴリズムが DHAM を決定することは容易にわかる．

計算量：頂点の順列のコードの長さは G のコードの長さ $\ell := |G|$ 以下である．頂点の個数を n とすると，$\log_2 n! = \sum_{k=1}^{n} \log_2 k \leq n \cdot \log_2 n \leq n^2$ により，$\log_2 n!$ は n^2 で抑えられる．G を行列でコード化すると ℓ は n^2 のオーダーとしてよいので，$\log_2 n!$ は ℓ のオーダーである．よって $n!$ は $2^{O(\ell)}$ で抑えられる．あとは上の例と同様にして，上記アルゴリズムの時間計算量が 2 の多項式乗で抑えられるとわかる．

3SAT および DHAM の計算コストについて，recursively enumerable 集合と似た性質が成り立つ．

例 **9.3** 任意の 3CNF（注）F に対して以下が成り立つ．

$$\begin{aligned} & F \in \mathrm{3SAT} \\ \Longleftrightarrow\ & \exists u \in \{0,1\}^* [u \text{ は真理値割り当て（のコード）であり，} u \text{ の下で } F \text{ は真}] \end{aligned}$$

注：3CNF とは３和積形命題論理式．ここでは，より詳しく言えばそのコード．

上記の [　] 内の条件は，$|u| \leq |F|$ という条件を付け加えても同値である．また，such that 以下の条件が成り立つかどうかは多項式時間のプログラムで判定できる．

例 **9.4** 任意の有向グラフ（のコード）G に対して以下が成り立つ．

$$\begin{aligned} & G \in \mathrm{DHAM} \\ \Longleftrightarrow\ & \exists u \in \{0,1\}^* \\ & \quad [u \text{ は } G \text{ の頂点の順列（のコード）であり，} u \text{ は } G \text{ のハミルトン閉路}] \end{aligned}$$

上記の [　] 内の条件は，$|u| \leq |G|$ という条件を付け加えても同値である．また，such that 以下の条件が成り立つかどうかは多項式時間のプログラムで判定できる．

上記の例を抽象化して以下の定義をする．

定義 **9.1** （非決定性多項式時間計算量クラス **NP** [AB2009, p.39, Def. 2.1]）言語 L が計算量クラス **NP** に属するとは，多項式 p と多項式時間のプログラム A が存在して，任意のビット列 x に対して以下が成り立つことをいう．

$$x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)} \text{ such that } \mathrm{A}(x,u) = 1.$$

3SAT と DHAM は **NP** に属する．$\mathbf{P} \subseteq \mathbf{NP}$ であることが容易にわかる．$\mathbf{P} \neq \mathbf{NP}$ であるかどうかは，有名な未解決問題である．

9.2 NP の特徴付け

[田中 1997, p.79]（あるいは [AB2009, pp.41–42]）と同様にして非決定性チューリング計算機を定義することができる．話を簡単にするため，与えられた状態とテープ記号に対し，可能な動作は高々二つであるとしよう．このように制限しても一般性を失わない．非決定性プログラムは，普通のプログラムに対し，1 ビットの理想的な乱数を取得する（公平なコイン投げをする）命令を付け加えたものに相当する．表が出たらこう，裏が出たらこう，というように次の動作を 2 通り用意できるのである．2 つの選択肢は同じでもよい．すなわち，コイン投げの結果を無視することも許される．

N が非決定性チューリング計算機であるとする．与えられた入力 x に対して，N の計算過程はこれまでのような列ではなく，二分木で表される．これを計算木といい，一本一本の枝を計算の道（path）という．同じ入力に対して，特定の道は 1（Yes）で終わるが別の道は 0（No）で終わることもありえる．一つでも 1（Yes）で終わる道が存在するとき，N は入力 x を受理するという[1]．多数決で受理を決める方式も研究されている．その場合，非決定性チューリング計算機ではなく，乱択チューリング計算機あるいは確率チューリング計算機という．

便宜上，計算木の高さを N の計算時間とよぶ．多項式時間のタイマーが付いた非決定性チューリング計算機を多項式時間非決定性チューリング計算機という．

$\mathbf{DTIME}(T(n))$ の定義における通常の（決定性）プログラムを非決定性チューリング計算機で置き換えることにより計算量クラス $\mathbf{NTIME}(T(n))$ を定める．このとき，以下が成り立つ．

定理 **9.2** [AB2009, p.41, Theorem 2.6]

$$\mathbf{NP} = \bigcup_{c \geq 1} \mathbf{NTIME}(n^c) = \bigcup_{p} \mathbf{NTIME}(p(n))$$

ここで p は多項式全体に渡って動くものとする．

したがって，L が **NP** に属することは以下と同値である：ある非決定性多項式時間チューリング計算機 N が存在して以下が成り立つ．

$$x \in L \Leftrightarrow \text{N は入力 } x \text{ を受理する．}$$

NP という名は，元々 nondeterministic polynomial-time に由来する．

[1] 第 I 部 計算可能性における「計算困難性の度合いの比較」の節で「決定性マシンが言語を受理する」という言葉を使った．今回は「非決定性マシンが入力を受理する」である．主語も目的語も異なるので注意．

9.3 coNP の定義

言語 L の補集合が **NP** に属するとき，L は **coNP** に属するという．**P** は補集合をとる操作について閉じているから，$\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{coNP}$ である．

例 **9.5** トートロジー全体の集合 TAUT は **coNP** に属する．

命題 **9.3** 言語 L に対して，以下の各条件は同値である．

1. L は **coNP** に属する．
2. 多項式 p と多項式時間のプログラム A が存在して，任意のビット列 x に対して以下が成り立つ．
$$x \in L \Leftrightarrow \forall u \in \{0,1\}^{p(|x|)} \mathrm{A}(x,u) = 1.$$
3. ある非決定性多項式時間チューリング計算機 N が存在して以下が成り立つ．
$$x \in L \Leftrightarrow \mathrm{N} \text{ は入力 } x \text{ を拒否する．}$$

9.4 多項式時間還元と完全性

二つの集合の複雑さを比較する尺度を再び導入する．

定義 **9.4** [渡辺 1992, 定義 6.1] A，B は $\{0,1\}^*$ の部分集合であるとする．

1. 次の条件をみたす関数 h を A から B への *polynomial-time many-one reduction*（[渡辺 1992, § 6.1] では polynomial time reduction）という．
   (a) h は $\{0,1\}^*$ から $\{0,1\}^*$ への全域的関数．
   (b) $\forall x \in \{0,1\}^* \ [x \in A \leftrightarrow h(x) \in B]$
   (c) h は多項式時間計算可能．
2. 上のような h が存在するとき，「A は B へ *polynomial-time many-one reducible* である」といい，$A \leq^{\mathrm{P}}_{\mathrm{m}} B$ と書く．

$\leq^{\mathrm{P}}_{\mathrm{m}}$ は擬順序 (pseudo ordering) である．すなわち，反射律と推移律をみたす．ただし反対称律はみたさない．

定義 **9.5** [渡辺 1992, 定義 6.2] $\mathcal{C}$ は計算量クラスであるとする．(1) 集合 A が $\forall L \in \mathcal{C} \ \ L \leq^{\mathrm{P}}_{\mathrm{m}} A$ という条件をみたすとき，A を（$\leq^{\mathrm{P}}_{\mathrm{m}}$ に関する）*$\mathcal{C}$-hard set*（$\mathcal{C}$ 困難集合）という．
(2) 集合 A が $\mathcal{C}$-hard かつ $A \in \mathcal{C}$ であるとき，A を（$\leq^{\mathrm{P}}_{\mathrm{m}}$ に関する）*$\mathcal{C}$-complete set*（$\mathcal{C}$ 完全集合）という．

定理 **9.6** (1) 3SAT，SAT，DHAM はいずれも NP-complete である．
(2) TAUT は coNP-complete である．

9.5　多項式時間確率計算量クラス PP

定義 9.7（多項式時間確率計算量クラス **PP** [AB2009, p.344–345, Def. 17.6]）言語 L が計算量クラス **PP** に属するとは，多項式 p と多項式時間（決定性）プログラム A が存在して，任意のビット列 x に対して以下が成り立つことをいう．確率は $u \in \{0,1\}^{p(|x|)}$ についての一様分布で考える．

$$x \in L \iff (\mathrm{A}(x,u)=1 \text{ となる確率}) \geq \frac{1}{2}$$

上記定義は決定性プログラムを用いている．$p(|x|)$ ビットの乱数を取得するところまでマシンの機能に組み入れ，乱択マシンを用いて定義する流儀もある．

補題 9.8 与えられた言語 L に対し，以下の二つの条件は同値である．

(1) $L \in \mathbf{PP}$.

(2) 多項式 p と多項式時間（決定性）プログラム B が存在して，任意のビット列 x に対して以下が成り立つ．

$$x \in L \iff (\mathrm{B}(x,u)=1 \text{ となる確率}) > \frac{1}{2}$$

証明　(1)⇒(2): (1) を仮定する．定義 9.7 にある通りの多項式 $p(|x|)$ とマシン A をとる．このとき，特定の $u \in \{0,1\}^{p(|x|)}$ が選ばれる確率は $1/2^{p(|x|)}$ である．根源事象の確率が $1/2^{p(|x|)}$ だから，確率はその整数倍となる．よって以下が成り立つ．

$$x \in L \implies (\mathrm{A}(x,u)=1 \text{ となる確率}) \geq \frac{1}{2}$$
$$x \notin L \implies (\mathrm{A}(x,u)=1 \text{ となる確率}) \leq \frac{1}{2} - \frac{1}{2^{p(|x|)}}$$

あとは，確率を少しだけ水増ししてやればよい．具体的には以下の通りである．多項式時間（決定性）プログラム A′ を，以下が成り立つように作る．以下，p で $p(|x|)$ を表す．

$x \in L$ であるかどうかに関わりなく，ランダムに選んだ $u \in \{0,1\}^p$ に対し，$\mathrm{A}'(x,u)=1$ となる確率がちょうど $1/2 + 1/2^p$.

このようなマシンを設計するには，次のようにすればよい．長さ p のビット列 u が与えられたとき，u の先頭 $p-1$ ビットがすべて 0 のときは 1 を出力する．そうでないときは，u の最終ビットを出力して終了する．

ここで新たな多項式時間（決定性）プログラム B を以下のように作る．入力 x および，$u = u_1 \cdots u_{p+1}$ が与えられたとする．u の最終ビットが 0 のときは $\mathrm{A}(x, u_1 \cdots u_p)$ を出力する．そうでないときは $\mathrm{A}'(x, u_1 \cdots u_p)$ を出力する．条件付き確率を計算すると以下が成り立つことがわかる．確率は $u \in \{0,1\}^{p+1}$ に対する一様分布で考える．

$$x \in L \implies (\mathrm{B}(x,u) = 1 \text{ となる確率}) \geq \frac{1}{2} + \frac{1}{2^{p+1}}$$
$$x \notin L \implies (\mathrm{B}(x,u) = 1 \text{ となる確率}) \leq \frac{1}{2}$$

よって (2) が成り立つ．

(2)⇒(1): 上と同様にして，確率を少しだけ下げてやればよい． □

定理 9.9 [Pa1994, p.257, Theorem 11.3]

$$\mathbf{NP} \subseteq \mathbf{PP}$$

証明　補題 9.8 における (1)⇒(2) の証明と同じ方針を用いる．まず，$L \in \mathbf{NP}$ と仮定する．定義 9.1 により以下のような多項式時間（決定性）プログラム A がある．

$$x \in L \implies (\mathrm{A}(x,u) = 1 \text{ となる確率}) \geq 1/2^{p(|x|)}$$
$$x \notin L \implies (\mathrm{A}(x,u) = 1 \text{ となる確率}) = 0$$

決定性マシン B を次のように作る．入力 x および，$u = u_1 \cdots u_{p+1}$ が与えられたとする．u の最終ビットが 0 のときは $\mathrm{A}(x, u_1 \cdots u_p)$ を出力する．そうでないときは 1 を出力する．

条件付き確率を計算すると補題 9.8 (2) の条件が成り立つことがわかる．よって補題 9.8 により，L は **PP** に属する． □

確率計算量クラス，とくに **PP** の部分クラスである **BPP** は，疑似乱数や暗号の研究にも登場する．

参考文献

[AB2009] Arora, S. and Barak, B., *Computational complexity, a modern approach*. Cambridge University Press (2009).

[Pa1994] Papadimitriou, C., *Computational complexity*. Addison-Wesley (1994).

[田中 1997] 田中 尚夫, 「情報の数理 計算論理入門」. 裳華房 (1997).

[リュー 1986] C.L. リュー （成嶋・秋山 訳）, 「離散数学入門」, マグロウヒル (1986).

[渡辺 1992] 渡辺治, 「計算可能性・計算の複雑さ入門」. 近代科学社 (1992).

鈴木登志雄「計算可能性理論 第 II 部 計算複雑性」

2012 年 9 月 初版

2019 年 3 月 1 日 改訂版 2019.03.01

2020 年 4 月 1 日 改訂版 2020.04.01，12 月 13 日 revision a，2022 年 9 月 24 日 revison b，10 月 6 日 revison c，2023 年 2 月 24 日 revison d

2024 年 3 月 1 日 改訂版 2024.03.01