PCP定理とその証明

清水 伸高 (東京科学大学)

Contents

序文

このノートは, 計算量理論で 90 年代に証明された重要な結果である PCP 定理とその証明についての講義ノートである. 計算量 (computational complexity) とは, 問題を解くために必要な計算リソースの量 (例えば計算時間, 記憶領域のサイズ, 乱択や量子性の有無や量) を意味し, 計算量理論 (computational complexity theory) とはそれぞれの問題の計算量を明らかにするための理論である. PCP 定理とは, 判定問題 (Yes か No で答える問題) の検証に要する計算量に関する結果であり, 端的に言うと, ある命題が真であると主張する証明が文字列として与えられたとき, その証明を検証するためには, 通常, 全ての文字を見て確認する必要があるが, PCP 定理によれば, その証明の一部だけを見ることで, その命題が真であるか否かを確率的に検証することができるという驚くべき結果である. 例えば, ある実行列 A と実ベクトル b に対して線形方程式系 $Ax = b$ は解を持つ, という命題を考えてみよう. この命題が真であるならば実際に解の一つ x を証明として提示することができるが, その証明が正しいかどうかを検証するためには検証者は Ax を実際に計算し, その各成分が b と一致するかを確認する必要がある. ところが PCP 定理によれば, 巧妙に構成された証明 π を提示することにより, その証明 π 全ての文字を見ることなく, 99%の確率で正しく検証できるのである (ここでは確率的な検証, つまり検証者はランダムネスを用いた検証を行う設定を考える).

このように, 局所的な情報だけを使って全体の構造を推測できるという PCP 定理の性質は, 単に理論的に興味深いだけでなく, 誤り訂正符号の構成, 確率論的手法の脱乱択化, 最適化問題の近似率限界の導出など, 理論計算機科学において広大な応用を持つ.

Chapter 1

導入

この集中講義では PCP 定理と呼ばれる計算量理論の基本的な結果について解説し, その証明を与える. PCP 定理は 1998 年に Arora and Safra [AS98] and Arora, Lund, Motwani, Sudan, and Szegedy [ALMSS98] によって証明された. この証明は代数的な手法に基づく誤り訂正符号を技巧的に組合せたものであり, 難解なものであったが, その後 Dinur [Din07] によってより簡潔な証明が与えられた. この講義では Dinur [Din07] による比較的簡単な証明を紹介する. ちなみに Dinur はのちにこの業績によりゲーデル賞を受賞している.

1.1 計算量理論の復習

まずは計算量理論のどの教科書にも載っているような基礎的な用語の定義を与える. これらの用語に明るい読者はセクション 1.2.2 から読み始めても良い. なお, このノートではアルゴリズムの定義 (チューリング機械の定義) は省略し, アルゴリズムについて述べる際は具体的な計算の手続きを述べる[1].

まずは基本的な記号の定義を与える:

- オーダー記法: 二つの関数 $f, g\colon \mathbb{N} \to \mathbb{N}$ に対し, $f(n) = O(g(n))$ であるとは, ある定数 $c > 0$ が存在して, 十分大きな全ての $n \in \mathbb{N}$ に対して $f(n) \leq cg(n)$ が成り立つことをいう. また, $f(n) = \Omega(g(n)), f(n) = o(g(n)), f(n) = \omega(g(n))$ なども同様に ($n \to \infty$ として) 定義する.

- 自然数 $n \in \mathbb{N}$ に対して $[n] = \{1, \ldots, n\}$ とする.

- 有限集合 S に対し, $x \sim S$ と書いたとき, x は S から一様ランダムに選ばれた元であることを意味する.

- 集合 S 上のベクトル $x \in S^n$ および $I \subseteq [n]$ に対し, $x_I \in S^I$ を, x の I への制限, すなわち, $x_I(i) = x(i)$ $(i \in I)$ と定義する.

[1] ひとまず Python や C 言語などで実装されたプログラムを考えれば良い. ただし, 計算機内では全ての数値は有限桁の二進数で表記されており, その読み書きや演算には少なくとも桁数に比例した計算時間がかかる. なお, $\sqrt{2}$ といった無理数は本来は有限桁で打ち切った近似値を扱うが, そのような小数はこの講義では扱わず, 特に断りのない限りは整数値のみを考える. また, 記憶領域へのアクセスは定数時間で行えると仮定する (チューリング機械であればテープの移動にかかる時間も考慮する).

- $\{0,1\}^* = \bigcup_{n\in\mathbb{N}}\{0,1\}^n$ を有限長の二進文字列全体とする.
- $x \in \{0,1\}^*$ に対して $|x|$ を x の文字数とする.
- アルゴリズム A に対し, $A(x)$ を入力 $x \in \{0,1\}^*$ に対するアルゴリズム A の出力とする. ここで, 単に「アルゴリズム」と言った場合は決定的アルゴリズムを指し, 乱択アルゴリズムについては明示的に言及する.
- 関数 $T(n)\colon \mathbb{N} \to \mathbb{N}$ を考える. 十分大きな全ての $n \in \mathbb{N}$ と全ての $x \in \{0,1\}^n$ に対して, A が $A(x)$ を出力するまでにかかる計算ステップ数の最大値が高々 $T(n)$ であるとき, アルゴリズム A の計算量は $T(n)$ であるという. 特に, ある (n に依らない) 定数 $c > 0$ が存在して計算量が $O(n^c)$ で抑えられるアルゴリズムを**多項式時間アルゴリズム**という[2]
- 計算量理論において慣例的に用いられる記法だが, 関数 $f(n)$ が n に関する多項式であることを $f(n) = n^{O(1)}$ と表す. 例えば多項式時間アルゴリズムとは, 十分大きな全ての $n \in \mathbb{N}$ に対して, 長さ n の任意の文字列 $x \in \{0,1\}^n$ を受け取ったときの時間計算量が $n^{O(1)}$ で抑えられるアルゴリズムである.
- 二つの文字列 $x, y \in \{0,1\}^*$ を入力として受け取るアルゴリズムは $A(x,y)$ と表す. 三つ以上の場合も $A(x,y,z)$ などと表す.

計算量理論で最も基本的な問題群として判定問題と呼ばれる問題群がある.

定義 1.1.1 (判定問題)

部分集合 $L \subseteq \{0,1\}^*$ を**判定問題 (または言語)** といい, アルゴリズムに与える L の入力を**インスタンス**という. また, インスタンス $x \in \{0,1\}^*$ は $x \in L$ であるとき, 判定問題 L の **Yes インスタンス**といい, そうでない場合は **No インスタンス**という.
文字列 $x \in \{0,1\}^*$ に対し $L(x) \in \{0,1\}$ を, $x \in L$ かどうかの指示関数, すなわち $x \in L$ ならば $L(x) = 1$, そうでなければ $L(x) = 0$ と定義する.
アルゴリズム A は, 任意の $x \in \{0,1\}^*$ に対して $A(x) = L(x)$ が成り立つとき, A は L を解くという.

例 1.1.2 (グラフ連結性判定問題)

グラフ $G = (V,E)$ の隣接行列を $A \in \{0,1\}^{|V|\times|V|}$ とする. この行列を長さ $|V|^2$ の二進文字列として表現したものを $\mathsf{str}(A)$ とする. すなわち, $\mathsf{str}(A)$ の第 k ビットは, $i = \lceil k/|V| \rceil + 1, j = k \mod |V| + 1$ に対して $A(i,j)$ の値として与えられる. このとき,

$$L = \left\{ \mathsf{str}(A) \in \{0,1\}^{|V|^2} \colon A \text{ は連結グラフ } G \text{ の隣接行列} \right\}$$

は与えられたグラフが連結であるかどうかを判定する判定問題である.

[2] 計算モデルによって一つ一つの計算ステップの定義は異なるが, 原理的には 1bit の演算や記憶領域への読み書きの回数と思えばよい. 多項式時間で動くかどうかの議論であれば, 多くの古典的な計算モデルは等価である.

この講義では「効率的に解ける」といった場合, 多項式時間アルゴリズムによって解けることを指す. そのような判定問題の集合をクラス P という.

定義 1.1.3 （クラス P）

判定問題 L は, それを解く多項式時間アルゴリズムが存在するとき, P に属するといい, $L \in \mathsf{P}$ と表す.

注釈 1.1.4 （入力のフォーマット）

例 1.1.2 では入力としてグラフの隣接行列を二進文字列として表現したものを考えたが, 一般にグラフの表現方法は他にも隣接リストなどが考えられる. しかし, 例えばグラフを隣接行列で表現するか隣接リストで表現するかは, それぞれのフォーマット間の変換が多項式時間で行えるため, **多項式時間で解けるかどうか**という点では等価である.
そのため, 厳密には問題を定義する際はその入力のフォーマット (グラフを隣接行列で表現するか, 隣接リストで表現するか, など) も指定する必要があるが, フォーマット間の変換が自明に多項式時間で行える場合に限ってはこの講義ではそのようなフォーマットの指定を省略し, 例えばグラフ連結性判定問題を「グラフが与えられたときにそれが連結であるかどうかを判定する問題」と表現する.

次に乱択アルゴリズムについて定義する. 端的に言えばアルゴリズムの内部でコイントスを行うものを乱択アルゴリズムという. ここでは明示的にランダムシードを受け取るアルゴリズムを乱択アルゴリズムと呼ぶことにする.

定義 1.1.5 （乱択アルゴリズム）

入力 $x \in \{0,1\}^*$ とは別にランダムシード (乱数表) と呼ばれる別の文字列 $s \in \{0,1\}^*$ を受け取るアルゴリズムを**乱択アルゴリズム**といい, ランダムシードであることを強調するために $A(x;s)$ などと表す. なお, 任意の $x, s \in \{0,1\}^*$ に対して $A(x;s)$ は有限時間で停止するとし, その計算量は $|x|$ のみに依存する関数で表せるとする. このとき, ランダムシード s の長さを常に A の計算量で上から抑える. すなわち, A の計算量が $T(n)$ であるとき, 十分大きな全ての $n \in \mathbb{N}$ と全ての $x \in \{0,1\}^n$ に対して A が読み込む s の文字数は高々 $T(n)$ であるため, $s \in \{0,1\}^{T(n)}$ であると仮定する. しばし, ランダムシード s を明記する必要が特にない場合は $A(x)$ と表す.
乱択アルゴリズムのランダムシードに関する確率, 期待値, 分散を議論する際は記号として $\Pr_A[\cdot], \mathbb{E}_A[\cdot], \mathbf{Var}_A[\cdot]$ を用いる. 乱択アルゴリズム A が判定問題 L を解くとは,

$$\Pr_A[A(x) = L(x)] \geq 2/3$$

が成り立つことをいう.

また, 入力とは別に文字列へのオラクルアクセスを受け取るアルゴリズムを考える.

定義 1.1.6 （オラクルアルゴリズム）

文字列 $\pi \in \{0,1\}^*$ に対し, π への**オラクルアクセス**を持つアルゴリズム $A^\pi(x)$ とは, 計算途中で π の指定された位置の文字を読むことができるアルゴリズムである. すなわち, π の i 番目の文字を読む操作を $A^\pi(x)$ の計算過程中に $O(\log|\pi|)$ 時間で行うことができるアルゴリズムである.[a] 同様に乱択オラクルアルゴリズムについても定義できる.

[a] 自然数 $i \in [|\pi|]$ を指定するために $O(\log|\pi|)$ ビットを定めなければならないため, $O(\log|\pi|)$ 時間を仮定している.

1.2 検証の計算量

数学全般における検証とは, ある命題が真であると主張する証明が与えられたとき, その証明が実際にその命題を正しく証明しているかどうかを確認することを意味する. 論文や記述試験の証明の査読や採点をイメージしてもらうとわかりやすいだろう. 計算量理論では検証やその計算量の議論は重要な研究テーマであり, その検証に要する計算量が議論される.

1.2.1 効率的な検証とクラス NP

判定問題 L と入力 $x \in \{0,1\}^*$ を与えられたとき, $x \in L$ かどうかを審議したい. ここで $x \in L$ を主張する証明が文字列 $\pi \in \{0,1\}^*$ で与えられたとする. このとき, **検証者**と呼ばれるアルゴリズムは x と π を読み込んで $x \in L$ かどうかを判定する. この判定を多項式時間で行えるとき, その判定問題 L の集合を NP という.

定義 1.2.1 （クラス NP）

判定問題 L は, 以下を満たす多項式時間アルゴリズム V と多項式 $p\colon \mathbb{N} \to \mathbb{N}$ が存在するとき, L は NP に属するという: アルゴリズム V は入力として $x, \pi \in \{0,1\}^*$ を受け取り, 0 または 1 を出力する.

1. もし $x \in L$ ならば, ある $\pi \in \{0,1\}^{p(|x|)}$ が存在して $V(x,\pi) = 1$ となる.
2. もし $x \notin L$ ならば, 全ての $\pi \in \{0,1\}^{p(|x|)}$ に対して $V(x,\pi) = 0$ となる.

また, このようなアルゴリズム V を **NP 検証者**といい, π を **NP 証拠**という.

注釈 1.2.2 （クラス P と NP の関係）

判定問題 L が P に属するならば $L \in$ NP である. 実際, 受け取った $x \in \{0,1\}^*$ に対して $L(x)$ を計算してそれを出力する検証者を考えばよい. すなわち P $\subseteq$ NP である. 一方, 逆側の包含関係 NP $\subseteq$ P が成り立つかどうかは P vs NP 問題と呼ばれる計算量理論における最も重要な未解決問題であり, 多くの研究者は NP $\subseteq$ P が成り立たないと信じている.

例 1.2.3 （合成数判定問題）

判定問題 $L = \{x \in \{0,1\}^* : x \text{ は合成数 }\}$ を考える．このとき, 検証者は x と π を読み込んで, $\pi \notin \{1, x\}$ かつ π が x を割り切るかどうかを判定する．もしも $x \in L$ である場合, 合成数なので非自明な約数を証拠 π として与えれば $V(x,\pi) = 1$ となる．そうでない場合, 非自明な約数は存在しないため必ず $V(x,\pi) = 0$ となる．このアルゴリズム V は入力長 (つまり数値の二進表現したときのビット長) に関する多項式時間で動作するため, L は NP に属する．

例 1.2.4 （グラフ彩色問題）

自然数 $k \geq 2$ とグラフ $G = (V, E)$ に対し, 関数 $c\colon V \to [k]$ が全ての辺 $\{u, v\} \in E$ に対して $c(u) \neq c(v)$ を満たすとき, c を G の k-彩色といい, k-彩色が存在するようなグラフは k-彩色可能であるという．任意の $k \geq 2$ に対し, 判定問題

$$L = \{G \in \{0,1\}^* : G \text{ は } k\text{-彩色可能 }\}$$

は NP に属する．検証者は G と π を読み込んで, π が G の k-彩色であるかどうかを判定する．もしも $G \in L$ である場合, G は k-彩色可能であるため, k-彩色の証拠 π を与えれば $V(G,\pi) = 1$ となる．そうでない場合, G は k-彩色可能でないため必ず $V(G,\pi) = 0$ となる．このアルゴリズム V は多項式時間で動作するため, L は NP に属する．

演習問題 1 （素数判定問題）

自然数 $n \in \mathbb{N}$ に対し, 判定問題 $\mathrm{PRIMES} = \{a \in \mathbb{N} : a \text{ は素数 }\}$ を考える．この問題は P に属することが知られている [AKS04] が, その複雑なアルゴリズムを用いずに初等的に $\mathrm{PRIMES} \in \mathsf{NP}$ を示したい．そのために, 以下の事実を用いる:
任意の自然数 $a \in \mathbb{N}$ と $\gamma \in \{1, \ldots, a-1\}$ に対し, $\gamma^0, \gamma^1, \ldots \pmod a$ は周期的である．さらに, 以下が成り立つ:

- a が素数であるならば, ある $\gamma \in \{1, \ldots, a-1\}$ が存在して $\gamma^0, \gamma^1, \ldots \pmod a$ の周期が $a-1$ である[a].
- 一方, a が素数でないならば, 全ての $\gamma \in \{1, \ldots, a-1\}$ に対して $\gamma^0, \gamma^1, \ldots \pmod a$ の周期は $a-1$ 未満である．特に, その周期 L は $a-1$ を割り切る．

これらの事実を用いて, 以下の小問に答えよ．

1. 次の検証者 V_1 を考える: 入力 $a \in \mathbb{N}$ と証拠 $\gamma \in \{1, \ldots, a-1\}$ に対し, $\gamma^0, \gamma^1, \ldots, \gamma^{a-2} \pmod a$ を全て検証し, これら全て相異なるかどうかを判定する．この検証者 V_1 が多項式時間アルゴリズム**でない**理由を簡潔に説明せよ．
2. 入力 $a \in \mathbb{N}$ に対し, a が素数であることの証拠として, 原始元 $\gamma \in \{1, \ldots, a-1\}$ および $a-1$ の素因数分解 $a - 1 = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$ および各 p_i が素数であることの証拠を再帰的に与える．この証拠を用いて, 検証者 V_2 は a が素数であるかどうかを多項式時間で判定できることを示せ．

[a] このような γ を原始元という．

1.2.2 局所的な検証とクラス PCP

一般に $x \in L$ かどうかの検証では, 証明 π の全ての文字を読む必要がある. しかし, 証明 π のうちの一部の文字を読むだけで $x \in L$ かどうかを**確率的に**判定できる場合がある. そのような性質を持つ証拠を**確率的検証可能な証拠** (Probabilistically Checkable Proof, PCP) という.

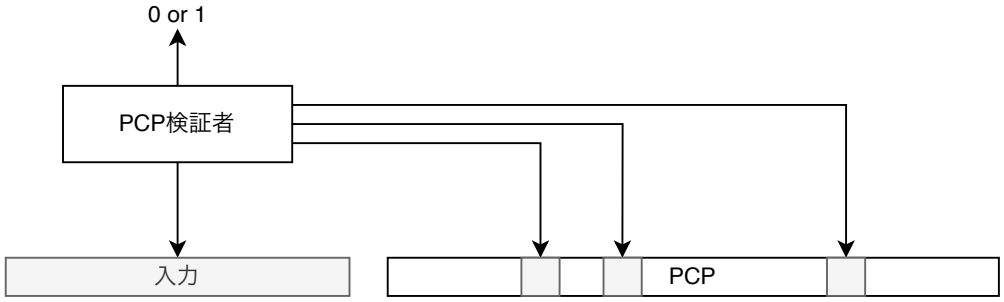

図 1.1: 確率的検証可能な証拠の概念図. 検証者は乱数を用いて証拠 π のうちの一部の文字のみを読み, それに基づいて判定を行う.

定義 1.2.5 (確率的検証可能な証拠)

二つの関数 $r, q\colon \mathbb{N} \to \mathbb{N}$ に対し, $\mathsf{PCP}(r, q)$ を以下の性質を持つ判定集合 L の集合とする: ある多項式時間オラクル乱択アルゴリズム V が存在して, 任意の $x \in \{0,1\}^*$ に対し,

1. もし $x \in L$ ならば, ある $\pi \in \{0,1\}^*$ が存在して, (V の乱択に関して) 確率 1 で $V^\pi(x) = 1$ となる (**完全性**).
2. もし $x \notin L$ ならば, 全ての $\pi \in \{0,1\}^*$ に対して, (V の乱択に関して) 確率 $1/3$ 以上で $V^\pi(x) = 0$ となる (**健全性**).
3. さらに, 入力長が $n = |x|$ のとき, $V^\pi(x)$ はオラクル π のうち高々 $q(n)$ 個の文字を読み, そのランダムシード長は $r(n)$ で抑えられる.

このようなオラクル乱択アルゴリズム V を **PCP 検証者**といい, 証拠 π を **PCP** という.

注釈 1.2.6 (PCP の長さ)

$\mathsf{PCP}(r, q)$ の証拠 π の長さは $q(n)2^{r(n)}$ で抑えられる. 各ランダムシード $s \in \{0,1\}^{r(n)}$ に対して検証者は π のうち高々 $q(n)$ 個の文字を読むため, 全てのランダムシードを列挙すると, アクセスされる可能性のある π の文字数は高々 $q(n)2^{r(n)}$ で抑えられる.

一般にランダムシード長 $r(n)$ と読み込む文字数 $q(n)$ が小さいほど良い PCP 検証者であると考えられる. PCP 検証者の構成は非常に難しい. 例えば例 1.2.4 のグラフ彩色問題に対する次の検証者を考えてみよう: 入力としてグラフ $G = (V, E)$ と証拠として関数 $\pi\colon V \to [k]$ を受け取り, この関数 π が実際に G の k-彩色であるかどうかを判定する. 検証者は G の辺 $\{u, v\} \in E$ をランダムに一つ選び, $\pi(u) \neq \pi(v)$ であるかどうかによって判定する. この検証

者は π のうち高々 $q(n) = O(1)$ 個の文字を読み, そのランダムシード長は $r(n) = O(\log n)$ で抑えられる. しかし, この検証者は健全性の条件を満たさない. 実際, G が k-彩色可能でない場合, どのような関数 $\pi\colon V \to [k]$ を与えても, 少なくとも一つの辺 $\{u, v\} \in E$ が存在して $\pi(u) = \pi(v)$ となるが, 検証者がこのような辺を引き当てる確率は最悪の場合, $1/|E|$ となるからである.

PCP 定理とは, ある $r(n) = O(\log n)$, $q(n) = O(1)$ に対して $\mathsf{PCP}(r, q) = \mathsf{NP}$ が成り立つことを主張する定理である. 例えばグラフ彩色問題は NP に属するため, 実は全段落の $r(n), q(n)$ を達成する PCP 検証者が存在するのである!

定理 1.2.7 （PCP 定理）

ある $r(n) = O(\log n)$, $q(n) = O(1)$ に対して $\mathsf{PCP}(r, q) = \mathsf{NP}$ が成り立つ.

注釈 1.2.8 （片側の包含関係）

PCP 定理において, $\mathsf{PCP}(r, q) \subseteq \mathsf{NP}$ は容易に示すことができる. 実際, 注釈 1.2.6 により, 証拠 π の長さは $q(n)2^{r(n)} = n^{O(1)}$ で抑えられる. また, $r(n) = O(\log n)$ より, 検証者は $2^{r(n)} = n^{O(1)}$ 個のランダムシードを列挙し, それら全てに対して PCP 検証者を適用し, その出力値の多数決をとることで, 入力 x に対して $x \in L$ かどうかを多項式時間で判定できる. PCP 定理の証明の本質的な難しさは逆側の包含関係 $\mathsf{NP} \subseteq \mathsf{PCP}(r, q)$ の証明にある.

1.2.3 3 彩色問題の NP 完全性

クラス NP に属する問題のうち最も難しいという性質を **NP 完全性**という. 本来は NP 完全性を定義するには帰着の概念を導入しなければならないが, この講義では 3 彩色問題が NP 完全であるという事実のみを用いるため, この事実について証明なしで述べる.

定理 1.2.9 （3 彩色問題の NP 完全性）

3 彩色問題 3COL は NP 完全である. すなわち, 任意の $L \in \mathsf{NP}$ に対して, 多項式時間アルゴリズム f が存在して, 任意の $x \in \{0,1\}^*$ に対して $x \in L$ と $f(x) \in \mathrm{3COL}$ は同値である.

つまり, 3COL を多項式時間アルゴリズムで解くアルゴリズムが存在するならば, NP に属する任意の判定問題 L を多項式時間で解くことができる. 実際, 定理 1.2.9 のアルゴリズム f を用いて L の入力を 3COL の入力に変換した後に, 3COL を解くアルゴリズムを適用すればよい. この意味で, 3COL は NP に属する問題の中で最も難しい問題 (の一つ) であるといえる.

注釈 1.2.10 （NP 完全性の定義）

この注釈は NP 完全性の定義を与えるものであり, この講義では必要ない. 二つの判定

問題 L, L' に対し, ある多項式時間アルゴリズム $f\colon \{0,1\}^* \to \{0,1\}^*$ が存在して,

$$x \in L \iff f(x) \in L'$$

を満たすとき, L は L' に**カープ帰着可能**といい, そのようなアルゴリズム f を**カープ帰着**という. このとき, L' を解く多項式時間アルゴリズムが存在すればそれを用いて L を解く多項式時間アルゴリズムを構成することができる. この意味で「L' は L 以上に難しい」とみなし, $L \leq_p L'$ と表す.
クラス NP に属するある判定問題 L は, 任意の $L' \in \mathsf{NP}$ に対して $L' \leq_p L$ が成り立つとき, NP **完全**であるという.

3 彩色問題の NP 完全性より, PCP 定理 (定理 1.2.7) を証明するには 3 彩色問題に対して PCP 検証者を構成すればよいことがわかる.

定理 1.2.11 （3 彩色問題の PCP 検証者）

ある $r(n) = O(\log n), q(n) = O(1)$ に対して, 3 彩色問題 3COL は $\mathsf{PCP}(r, q)$ に属する.

演習問題 2

定理 1.2.11 と定理 1.2.9 を仮定して, PCP 定理 (定理 1.2.7) を証明せよ.

1.3 PCP 定理の応用

PCP 定理の応用として, 様々な組合せ最適化問題に対する近似の NP 困難性を導出することができる.

1.4 PCP 定理の歴史

PCP 定理は 1992 年に Arora と Safra によって証明された. その後, 2005 年に Dinur によって証明が簡略化され, 2010 年に Arora と Barak によって証明が再構成された.

Chapter 2

制約充足問題

制約充足問題 (Constraint Satisfaction Problem, CSP) は, 計算量理論において重要な問題の一つであり, PCP 定理の証明においても中心的な役割を果たす.

2.1 制約充足問題の定義

制約充足問題とは端的に言えば連立方程式の解の存在性判定を問う判定問題である.

定義 2.1.1 （制約充足問題）

制約充足問題 (CSP) とは次の要素からなる組 $\varphi = (X, \Sigma, \mathcal{I}, \mathcal{C})$ を入力とする判定問題である:

- **アルファベット**: 有限集合 Σ.
- **変数列**: $X = (\mathrm{x}_1, \ldots, \mathrm{x}_n)$.
- 制約の引数の列: $\mathcal{I} = (I_1, \ldots, I_m)$. ここで, 各 I_i は $I_i \neq \emptyset$ かつ $I_i \subseteq [n]$ を満たす.
- **制約の列**: $\mathcal{C} = (c_I)_{I \in \mathcal{I}}$. ここで, 各 c_I は $c_I \subseteq \Sigma^{|I|}$ を満たす.

各 $\mathrm{x} \in X$ を**変数**, 各 $c_I \in \mathcal{C}$ を**制約**という.
入力 $(X, \Sigma, \mathcal{I}, \mathcal{C})$ は, ある変数への割り当て $a\colon X \to \Sigma$ が存在して, 任意の $I = \{i_1, \ldots, i_k\} \in \mathcal{I}$ (ただし $i_1 < \cdots < i_k$) について,

$$a(I) := (a(\mathrm{x}_{i_1}), \ldots, a(\mathrm{x}_{i_k})) \in c_I$$

であるとき, かつその時に限り Yes インスタンスである.
全ての $i \in [m]$ について $|I_i| \leq q$ であるとき, この CSP は **q-CSP** という.
また, 固定した割り当て $a\colon X \to \Sigma$ に対する φ の**不満足値**を

$$\mathrm{UNSAT}_\varphi(a) = \Pr_{I \sim \mathcal{I}}[a(I) \notin c_I]$$

とする. ここで $I \sim \mathcal{I}$ は I が $\mathcal{I}$ から一様ランダムに選ばれたことを意味する. さらに, 全ての割り当てに関して不満足値の最小値

$$\mathrm{UNSAT}(\varphi) = \min_{a \in \Sigma^X} \mathrm{UNSAT}_\varphi(a)$$

を φ の**不満足値**という.

各制約 $c_I \in \mathcal{C}$ は, その制約が充足される割り当ての集合を表す. 例えば $c_I = \Sigma^{|I|}$ であるならば, 任意の割り当てに対して c_I は充足されるし, $c_I = \emptyset$ であるならば, どの割り当ても c_I を充足しない.

例 2.1.2 （グラフ彩色問題）

グラフ彩色問題 (例 1.2.4) は 2-CSP である. 実際, グラフ $G = (V, E)$ に対して

- 変数列を $X = V$ とする.
- アルファベットを $\Sigma = [k]$ とする.
- 各制約 c_e は各辺 $e = \{u, v\} \in E$ に付随していて,

$$(a_u, a_v) \in c_e \iff a_u \neq a_v$$

と定義する.

このとき, グラフ G が k-彩色可能であることと, この CSP が Yes インスタンスであることは同値である.

2.1.1 PCP との関係

ランダムシード長 $r = r(n)$ かつクエリ数 $q = O(1)$ の PCP 検証者は $\Sigma = \{0, 1\}$ の場合の q-CSP によって表現でき, 逆に q-CSP は PCP 検証者として表現できる. 実際, q クエリの PCP 検証者 $V^\pi(x)$ を考えよう. 証明の長さを $|\pi| = \ell$ とする. 入力 x とランダムシード s を固定したときの $V^\pi(x; s)$ が証明中の読み込む文字のインデックスの集合を $I_s \subseteq [\ell]$ とする (ここで $|I_s| \leq q$). このとき, $V^\pi(x; s)$ は $\{0, 1\}^{I_s}$ を $\{0, 1\}$ に写す関数を定める. この関数を制約 c_s とみなすことで, 検証者 $V^\pi(x)$ は q-CSP のインスタンスとして表現できる. このとき, q-CSP のインスタンスの変数集合は証明 π に対応する. もし $x \in L$ であるならば, 確率 1 で $V^\pi(x) = 1$ となるような π が存在する. つまり, 全てのランダムシード s に対して $V^\pi(x; s) = 1$ となるような π が存在するため, 先ほど構成した q-CSP のインスタンスは Yes インスタンスである. 逆に $x \notin L$ であるならば, 全ての π に対して確率 $1/3$ 以上で $V^\pi(x) = 0$ となる. これは, q-CSP インスタンスに対して, 全ての割り当てを考えても, 全体の制約のうち少なくとも $1/3$ の割合は充足されない (すなわち, UNSAT の値が $1/3$ 以上となる) ことを意味する.

PCP 検証者	q-CSP
PCP π	割り当て
ランダムネスを固定した時の判定	CSP の制約
PCPπ を拒否する確率	割り当ての不満足値 UNSAT(π)

Table 2.1: PCP と CSP の対応関係

この対応関係に基づいて, PCP 定理を CSP を用いた言葉で表すことができる.

定理 2.1.3 （PCP 定理の CSP 版）

ある関数 $m = n^{O(1)}$, $q = O(1)$, $\ell = n^{O(1)}$, 定数 $c \in \mathbb{N}$, $\varepsilon > 0$, および次の性質を満たす多項式時間決定的アルゴリズム A が存在する: 3 彩色問題のインスタンス $G = (V, E)$ を入力として受け取り, G の頂点数を n としたとき, A は高々 $\ell(n)$ 個の変数と $m(n)$ 個の制約および要素数 c のアルファベットからなる q-CSP のインスタンス φ を出力する. さらにこのインスタンス φ は

- 入力 G が 3COL の Yes インスタンスであるとき, $\mathrm{UNSAT}(\varphi) = 0$ となる (すなわち φ は Yes インスタンス).
- 入力 G が 3COL の No インスタンスであるとき, $\mathrm{UNSAT}(\varphi) \geq \varepsilon$ となる.

補題 2.1.4

定理 2.1.3 と定理 1.2.11 は同値である.

証明. それぞれの方向を別々に証明する.

定理 2.1.3⇒ 定理 1.2.11 の証明. ある $r = O(\log n)$, $q' = O(1)$ に対して, 3COL に対するシード長 r, クエリ回数 q' の PCP 検証者 V^π を構成する. 入力としてグラフ $G = (V, E)$ を受け取り, 定理 2.1.3 のアルゴリズム A を用いて q-CSP のインスタンス φ を出力する. また, PCP π はこの q-CSP のインスタンス φ の割り当てとして解釈し, PCP 検証者 V^π は以下の操作を十分大きな定数回繰り返す: 一様ランダムに制約 c_i を選択し, その制約に含まれる変数に対する割り当て π の値を読み込み, この制約が充足**されない**ならば 0 を出力し終了する. 何度も繰り返した末に終了しなかったのであれば, 1 を出力して終了する. この検証者は繰り返しの回数が $O(1)$ であり, それぞれの繰り返しにおいては高々 q 個の変数の値を読み込むため, クエリ回数は $q' = O(q) = O(1)$ となる. なお, ここでアルファベットサイズ c が定数であることに留意する (実際には PCP π は二進文字列なので, 変数割り当てを読み込む際には $\log_2 c$ 文字を読み込んでいる). また, ランダムシードはランダムな制約を選ぶために使われてるため, その長さは $O(\log m) = O(\log n)$ となる.

もしグラフ G が Yes インスタンスならば, φ も Yes インスタンスであるため, π をその充足割り当てとすれば, 全ての制約 c_i が充足されるため, (制約の選び方のランダムネスに関して) 確率 1 で $V^\pi(G) = 1$ となる. もしグラフ G が No インスタンスならば, $\mathrm{UNSAT}(\varphi) \geq \varepsilon$ である. 従って, 任意の割り当て π に対して, 一様ランダムな制約 c_i が充足される確率は高々 $1 - \varepsilon$ である. よって, この操作を $\lceil 10/\varepsilon \rceil = O(1)$ 回繰り返すと, 少なくとも確率 $1/3$ で充足されない制約が一度以上選ばれ, 検証者は 0 を出力する.

定理 1.2.11⇒ 定理 2.1.3 の証明. 仮定より, 3COL に対する, ランダムシード長 $r = O(\log n)$, クエリ回数 $q = O(1)$ の PCP 検証者 V^π が存在する. アルゴリズム A は, 入力 G に対して, 全てのランダムシード $s \in \{0,1\}^r$ を列挙して, それぞれの $V^\pi(G; s)$ を関数 c_s とみなして, これらを制約とする CSP を出力する. 各 $V^\pi(G; s)$ は, π を変数とみなしたとき, 高々 q 個の変数の値を読み込むため, $(c_s)_{s \in \{0,1\}^r}$ は $2^r = n^{O(1)}$ 個の制約からなる q-CSP となる. なお, V^π は多項式時間アルゴリズムなので, A も多項式時間アルゴリズムである. □

従って, 以降は定理 2.1.3 の証明に注力する.

2.1.2 制約グラフ

PCP 定理の証明は, NP-完全な 2-CSP である 3 彩色問題のインスタンスからスタートし, このインスタンスを適当な q-CSP にうまく変換することによって与えられる. この変換の記述を容易にするために, 2-CSP のインスタンスをグラフとして表現する方法として, **制約グラフ**の概念を導入する.

定義 2.1.5 （制約グラフ）

2-CSP のインスタンス $\varphi = (X, \Sigma, \mathcal{I}, \mathcal{C})$ に対し, 以下で定まる組 $G = \langle (V, E), \Sigma, \mathcal{C}' \rangle$ を**制約グラフ**という[a]:

- (V, E) はグラフである. ただし頂点集合は $V = X$ であり, 辺集合は $E = \mathcal{I}$ である. なお, ここで考えるグラフは多重辺や自己ループを持ちうるもの[b]とし, 特に $|I_i| = 1$ の場合は対応する辺は自己ループとする.
- アルファベット Σ.
- 制約の列 $\mathcal{C}' = (c'_e)_{e \in E}$ は, $|e| = 2$ ならば $c'_e = c_e$ とし, $|e| = 1$ ならば $c'_e = \{(u, u) \in \Sigma^2 : u \in e\}$ とする. これにより, 全ての c'_e は Σ^2 の部分集合となる.

制約グラフ $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ および割り当て $a : V \to \Sigma$ に対して, その**不満足値**を

$$\mathrm{UNSAT}_G(a) = \Pr_{e \sim E}[a(e) \notin c_e]$$

と定義し (ここで $e = \{u, v\}$ $(u < v)$ に対して $a(e) = (a(u), a(v)) \in \Sigma^2$ とする), G の不満足値を

$$\mathrm{UNSAT}(G) = \min_{a \in \Sigma^V} \mathrm{UNSAT}_G(a)$$

と定義する.
制約グラフ $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ に対して, その**サイズ**を

$$\mathrm{size}(G) = |V| + |E|$$

と定義する.

[a] 制約グラフの表記に用いる括弧は形式的には本来 $((V, E), \Sigma, \mathcal{C}')$ のようにすべきだが, 可読性のためあえて外側の括弧を $\langle \cdot \rangle$ としている.
[b] 具体的には E は多重集合であり, 自己ループに対応する辺は $\{u\}$ と表す. 詳細は定義 2.2.1 を参照.

注釈 2.1.6 （入力長とサイズの関係）

本講義で考える制約グラフのアルファベットサイズ $|\Sigma|$ は常に定数, すなわち $|V|$ や $|E|$ に依存しない値であるとする. この仮定の下, 制約グラフ $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ を指定するために必要なビット数を考える. グラフ (V, E) は隣接行列で表現すると $|V|^2$ ビットで表現できる. 各辺 $e \in E$ に付随する制約 $c'_e \subseteq \Sigma^2$ は, 各 $(a, b) \in \Sigma^2$ について $(a, b) \in c'_e$

かどうかを表すビットを (a,b) について並べれば良いため, 全部で $|E|\cdot|\Sigma|^2$ ビットで表現できる. よって, 制約グラフ G は $|V|^2+|E|\cdot|\Sigma|^2=O(\mathrm{size}(G)^2)$ ビットで表現できる. このことから, 制約グラフを入力として受け取るアルゴリズムが多項式時間かどうかを議論する際は, その時間計算量が $\mathrm{size}(G)$ に関して多項式かどうかを議論すれば良いことになる.

2.2 多重グラフの導入とエクスパンダーグラフ

以後, 制約グラフに関する様々な操作をしていく上で多重グラフのフォーマルな定義を与えておく.

定義 2.2.1 （多重グラフ）

有限集合 V と多重集合 E の組 (V,E) を**多重グラフ**という. ここで E は $V\cup\binom{V}{2}$ の元から構成される多重集合であり, $e\in E$ は $|e|=1$ ならば**自己ループ**であるという.
二頂点 $u,v\in V$ の間の**重み**を

$$w(u,v)=|\{e\in E\colon e=\{u,v\}\}|$$

と定義し, 頂点 u の**次数**を

$$\deg(u)=\sum_{v\in V}w(u,v)$$

と定義する.[a] また, $W=(w(u,v))_{u,v\in V}$ を**重み行列**と呼び, $P(u,v):=\frac{w(u,v)}{\deg(u)}$ で定まる行列 $P\in[0,1]^{V\times V}$ を**遷移確率行列**という.

[a] 自己ループの次数への寄与は 1 であることに留意されたい (文脈によってはこの寄与が 1 である場合もある).

グラフの次数を並べたベクトル $d=(\deg(v))_{v\in V}\in\mathbb{R}^V$ を**次数ベクトル**という. 次数ベクトルは

$$d=W\mathbf{1}$$

で与えられる.
以下に正則グラフとエクスパンダーグラフの定義を述べる.

定義 2.2.2 （正則性とエクスパンダー性）

n 頂点の多重グラフ $G=(V,E)$ の全ての頂点の次数が d に等しいとき, G は d-**正則**であるという.
また, 遷移確率行列 P の固有値 $1=\lambda_1\geq\lambda_2\geq\cdots\geq\lambda_n\geq-1$ が

$$\lambda:=\max\{|\lambda_2|,|\lambda_n|\}\leq\lambda$$

を満たすとき, G は λ-**エクスパンダー**であるという.

多重グラフ G が正則ならば P は対称行列となるため実固有値を持つことが直ちに従うが, 一般の場合でも実固有値を持つことが示せる. さらに, Gershgorin の定理からそれらの固有値の絶対値は 1 以下であることが従う. また, 全成分 1 のベクトル $\mathbf{1}$ は P の固有値 1 に対する固有ベクトルとなるため, $\lambda_1 = 1$ である.

演習問題 3

任意の多重グラフ G の遷移確率行列 P は実固有値を持つことを示せ.

第一固有値 1 の多重度はグラフの連結成分の個数に等しいことが知られている. ここではその特殊ケースである以下の事実を用いる.

命題 2.2.3

多重グラフ G が連結であるならば, 遷移確率行列 P の固有値 1 の多重度は 1, すなわち $\lambda_2 < 1$ である.

2.2.1 正則エクスパンダーの性質

この節では正則かつエクスパンダー性を持つ単純グラフの性質は同様に多重グラフに対しても成り立つことを確認する. 特に, 正則性より遷移確率行列 P は対称となることに留意されたい.

補題 2.2.4 (エクスパンダー混交補題)

連結な頂点数 n の多重グラフ G が d-正則かつ λ-エクスパンダーであるとする. 二つの頂点部分集合 $S, T \subseteq V$ に対して

$$W(S,T) = \sum_{u \in S, v \in T} w(u,v)$$

とすると, 任意の $S, T \subseteq V$ に対して

$$\left| W(S,T) - \frac{d}{n}|S||T| \right| \leq \frac{\lambda d}{n} \sqrt{|S||T||V \setminus S||V \setminus T|}$$

が成り立つ. 特に, $|S| \leq n/2$ を満たす任意の $S \subseteq V$ に対して

$$W(S, V \setminus S) \geq (1-\lambda)\frac{nd}{2}$$

が成り立つ.

注釈 2.2.5 （直感的な意味）

簡単のため自己ループを持たないグラフを考える. このグラフが n 頂点 d-正則ならば全部で $nd/2$ 本の辺を持つ. 全部で $\binom{n}{2} \approx n^2/2$ 個の頂点対があるため, 辺密度は d/n である. さて, グラフの辺が $\binom{V}{2}$ に「均一に」散らばっていると仮定すると, 任意に固定した頂点部分集合 $S, T \subseteq V$ に対してその間をまたがる辺の本数 $W(S,T)$ はおよそ $(d/n) \cdot |S||T|$ であることが期待される. グラフ G がエクスパンダー性を持つ場合, この期待値からのずれの上からの評価を与えるのがエクスパンダー混交補題である.

証明. 「特に, ...」の部分は前半の主張に $T = V \setminus S$ を代入して $|V \setminus S| \geq n/2$ を用いれば示せるので, 前半の主張を証明する.

全成分 1 の行列 $J \in \mathbb{R}^{V \times V}$ に対し, $M := P - \frac{1}{n}J$ とおく. また, 頂点部分集合 $S \subseteq V$ に対し, $\mathbf{1}_S \in \mathbb{R}^V$ を

$$\mathbf{1}_S(u) = \begin{cases} 1 & \text{if } u \in S \\ 0 & \text{otherwise} \end{cases}$$

で定める. 二つのベクトル $x, y \in \mathbb{R}^V$ を,

$$x = \mathbf{1}_S - \frac{|S|}{n}\mathbf{1},$$
$$y = \mathbf{1}_T - \frac{|T|}{n}\mathbf{1}$$

とする. ベクトル x, y は $\mathbf{1}$ に直交するので $\mathbf{1}_S = x + \frac{|S|}{n}\mathbf{1}$ は $\mathbf{1}_S$ の直交分解となっていること, 及び $W\mathbf{1} = d\mathbf{1}$ に着目すると,

$$\begin{aligned} W(S,T) &= \mathbf{1}_S^T W \mathbf{1}_T \\ &= x^T W y + \frac{|S||T|}{n^2}\mathbf{1}^T W \mathbf{1} \\ &= x^\top W y + \frac{d}{n}|S||T| \end{aligned}$$

が成り立つ.

従って

$$\begin{aligned} \left| W(S,T) - \frac{d}{n}|S||T| \right| &= \left| x^\top W y \right| \leq \|x\|_2 \|Wy\|_2 && \because \text{Cauchy-Schwarz の不等式} \\ &\leq d\lambda \|x\|_2 \|y\|_2 && \because \text{レイリー商と固有値の関係} \\ &= \frac{\lambda d}{n}\sqrt{|S||T||V \setminus S||V \setminus T|} && \because \|x\|_2^2 = \frac{|S|(n-|S|)}{n} \end{aligned}$$

を得る. □

次にグラフの**べき乗**の操作を定義する.

定義 2.2.6 （グラフのべき乗）

重み行列 W を持つ多重グラフ $G = (V, E)$ および $k \geq 0$ に対し, 多重グラフ $G^k = (V, E^k)$ を

$$W' = W^k$$

を重み行列とする多重グラフと定める.

元のグラフ G の各二頂点 $u, v \in V$ に対し, uv 間の長さ k の路の個数だけ uv 間の辺を追加することで G^k を得られる.

補題 2.2.7 （正則エクスパンダー性のべき乗）

頂点数 n の d-正則かつ λ-エクスパンダーである多重グラフ G が与えられたとする. このとき, G^k は d^k-正則かつ λ^k-エクスパンダーである.

証明. べき乗で得られるグラフ G^k の重み行列は $W' = W^k$ であるため, その次数ベクトルは

$$W'\mathbf{1} = W^k\mathbf{1} = d^k\mathbf{1}$$

となるため, G^k は d^k-正則である.

さらに, G^k の遷移確率行列は P^k で与えられるため, $1 = \lambda_1 \geq \cdots \geq \lambda_n \geq -1$ に対して P^k の固有値は $1 = \lambda_1^k \geq \cdots \geq \lambda_n^k \geq -1$ となる. 今, $\max\{|\lambda_2|, |\lambda_n|\} \leq \lambda$ であるため, $\max\{|\lambda_2^k|, |\lambda_n^k|\} \leq \lambda^k$ である. よって, G^k は λ^k-エクスパンダーである. □

2.2.2 エクスパンダーグラフの構成

エクスパンダーグラフは, その構造から様々な応用が知られている. 特に, 各 $n \geq 2$ に対して頂点数 n のエクスパンダーグラフを $n^{O(1)}$ 時間で構成できることが知られている.

定理 2.2.8 （エクスパンダーグラフの構成）

ある定数 $d_0 \geq 3$, $\lambda_0 < 1$ および以下を満たす多項式時間アルゴリズム A が存在する: アルゴリズム A は入力として $1^n = (\underbrace{1, \ldots, 1}_{n})$ を受け取り, 頂点数 n の d_0-正則かつ λ_0-エクスパンダーグラフの隣接行列 W を出力する.

定理 2.2.8 の証明はエクスパンダーグラフの構成に関するブレイクスルーの結果 [RVW02] の結果に軽微な修正を施すことによって得られる.

定理 2.2.9 （エクスパンダーグラフ族の構成）

ある定数 $d_0' \in \mathbb{N}$, $\lambda_0' < 1$ および以下を満たす多項式時間アルゴリズム A' が存在する: 各 $k \in \mathbb{N}$ に対して 1^{2^k} を入力として受け取り, 頂点数 2^k の d_0'-正則かつ λ_0'-エクスパンダーグラフの隣接行列 W' を出力する.

定理 2.2.9 の証明はジグザク積と呼ばれるグラフの積に関する手法を用いることで得られるが, 本講義のスコープからは逸脱するので割愛する. 定理 2.2.8 の証明, すなわち一般の頂点数のエクスパンダーグラフを得るには, まず $2^{k-1} < n \leq 2^k$ を満たす k に対して定理 2.2.9 を適用し, その後にそのグラフが n 頂点になるようにうまく二つの頂点を縮約し, 必要に応じて頂点に自己ループまたは多重辺を追加することによって正則にすることによって得られる.

なお, 補題 2.2.7 より, 次数を大きくすることによって固有値 λ_0 をいくらでも 0 に近づけることが可能である.

Chapter 3

ギャップ増幅補題

Dinur による PCP 定理の証明は, 与えられた制約グラフの不満足値を段階的に増幅していくアプローチに基づく. その中核を担うのがこのギャップ増幅補題であり, 制約グラフの不満足値を増幅できることを保証する補題である. 本チャプターはこの補題の証明を与える.

3.1 主張

ギャップ増幅補題は以下のように述べられる.

補題 3.1.1 （ギャップ増幅補題）

二つの定数 $c > 0, \alpha \in (0,1)$, アルファベット Σ, および以下の性質を満たす決定的多項式時間アルゴリズム A が存在する: アルゴリズム A は制約グラフ $G = \langle (V,E), \Sigma, \mathcal{C} \rangle$ を入力として受け取り, 次の性質を満たす別の制約グラフ $G' = \langle (V',E'), \Sigma, \mathcal{C}' \rangle$ を出力する:

- $\mathrm{size}(G') \leq c \cdot \mathrm{size}(G)$.
- $\mathrm{UNSAT}(G) = 0$ ならば $\mathrm{UNSAT}(G') = 0$.
- $\mathrm{UNSAT}(G) > 0$ ならば $\mathrm{UNSAT}(G') \geq \min\{\alpha, 2 \cdot \mathrm{UNSAT}(G)\}$.

PCP 定理 (定理 2.1.3) の証明は, この補題を繰り返し適用することによって得られる.

補題 *3.1.1* の下での定理 *2.1.3* の証明. 3 彩色問題のインスタンスを入力として受け取り, その制約グラフを G_0 とし, 頂点数を n とする. この制約グラフは単純グラフであるため, $\mathrm{UNSAT}(G_0) = 0$ もしくは $\mathrm{UNSAT}(G_0) \geq \frac{1}{n^2}$ である. 補題 3.1.1 のアルゴリズムを A とし, 各 $i = 1, \ldots, \lceil 2\log_2 n \rceil$ について, 制約グラフ G_i を $G_i = A(G_{i-1})$ として定義し, 最終的に得られる制約グラフを $G' = G_{\lceil 2\log_2 n \rceil}$ とする. 各 G_i のサイズは $\mathrm{size}(G_i) \leq c \cdot \mathrm{size}(G_{i-1})$ であるため, $\mathrm{size}(G') \leq c^{\lceil 2\log_2 n \rceil} \cdot \mathrm{size}(G_0) = \mathrm{size}(G_0)^{O(1)}$ である. 従って G' は多項式時間で構成できる.

また, 不満足値については, 以下のようになる:

- もしも G_0 が Yes インスタンスであるならば, 全ての i に対して G_i も Yes インスタンスであり, 特に $\mathrm{UNSAT}(G') = 0$ である.

- もしも G_0 が No インスタンスであるならば, $\mathrm{UNSAT}(G_0) \geq \frac{1}{n^2}$ かつ $\mathrm{UNSAT}(G_i) \geq \min\{\alpha, 2 \cdot \mathrm{UNSAT}(G_0)\}$ であるため, $\mathrm{UNSAT}(G') \geq \min\left\{\alpha, 2^{2\log_2 n} \cdot \frac{1}{n^2}\right\} = \alpha$ である.

□

ギャップ増幅補題では与えられた制約グラフを変換していく. 表記の簡略化のため, グラフに対する性質を表す用語を制約グラフにもそのまま適用することとする. 例えば (V, E) が連結であるときに $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ は連結であるという. また, (V, E) が正則であるときに $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ は正則であるという.

3.2 制約グラフの定数次数エクスパンダー化

まず, 与えられた制約グラフを, 不満足値をそれほど減らさずに定数次数の正則性かつエクスパンダー性を持つように変形する.

補題 3.2.1 （定数次数エクスパンダー化）

ある定数 $\lambda < 1$, $d \in \mathbb{N}$, $c > 0$, $\beta > 0$ が存在して, 任意の制約グラフ $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ を入力として受け取り, 以下の性質を満たす別の制約グラフ $G' = \langle (V', E'), \Sigma, \mathcal{C}' \rangle$ を出力する決定的多項式時間アルゴリズム A が存在する:

- G' は自己ループを持つ d-正則 λ-エクスパンダーである.
- $\mathrm{size}(G') \leq c \cdot \mathrm{size}(G)$.
- $\mathrm{UNSAT}(G) = 0$ ならば $\mathrm{UNSAT}(G') = 0$.
- $\mathrm{UNSAT}(G') \geq \beta \cdot \mathrm{UNSAT}(G)$.

この補題の証明は次数の削減とエクスパンダー化の二つのステップからなる.

3.2.1 次数の削減

まず, 与えられた制約グラフを定数次数の正則グラフに変換する補題を示す. この変換によって UNSAT の値は定数倍しか変化しない.

補題 3.2.2 （次数削減補題）

ある定数 $d \in \mathbb{N}$, $c > 0$ が存在して, 任意の制約グラフ $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ を入力として受け取り, 以下の性質を満たす別の制約グラフ $G' = \langle (V', E'), \Sigma, \mathcal{C}' \rangle$ を出力する決定的多項式時間アルゴリズム A_1 が存在する:

- G' は d-正則である.
- $|V'| \leq 2|E|$.
- $\mathrm{UNSAT}(G) = 0$ ならば $\mathrm{UNSAT}(G') = 0$.
- $\mathrm{UNSAT}(G') \geq c \cdot \mathrm{UNSAT}(G)$.

証明. 与えられた制約グラフを $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ とし, 変換によって得られる制約グラフを $G' = \langle (V', E'), \Sigma, \mathcal{C}' \rangle$ とする. フォーマルな構成を与える前に, まず図例を先に示す.

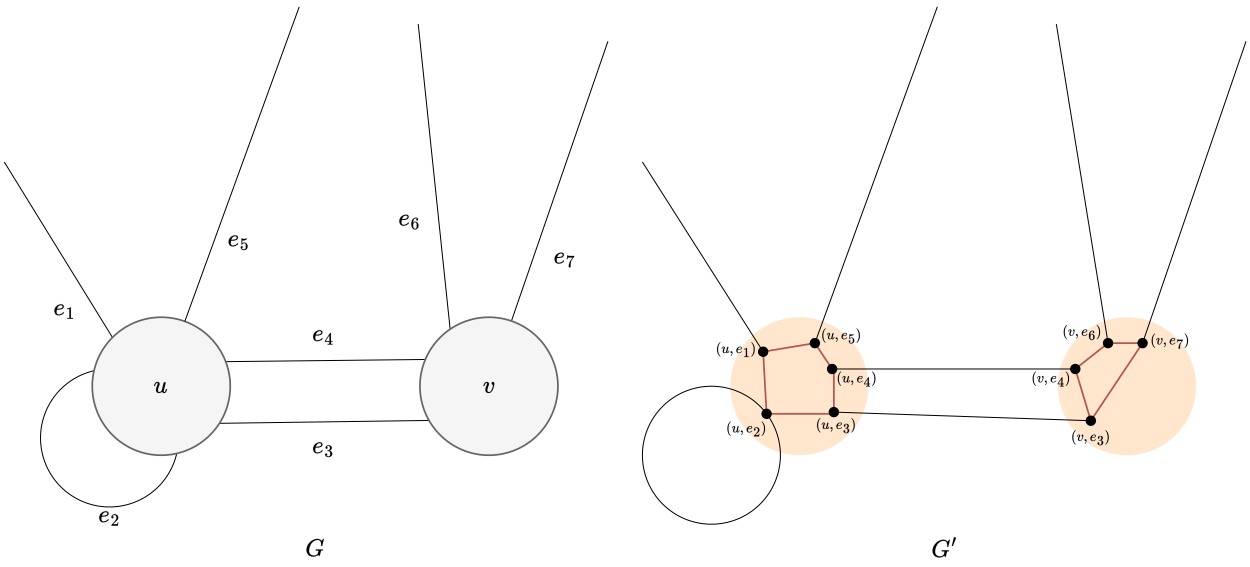

図 3.1: 次数削減変換の例. 橙色の内部の頂点集合がクラウドであり, クラウド内の辺は定理 2.2.8 によって構成されるが, ここでは図の簡単のため X_d を長さ d の閉路としている. この例の場合, G' は 3-正則となる.

元のグラフの各頂点 $u \in V$ に対し,

$$[u] = \{(u, e) \in \{u\} \times E \colon e = \{u, v\} \text{ for some } v \in V\}$$

を (この証明のローカルな用語として)u-**クラウド**と呼ぶことにする. 新しい頂点集合 V' は $V' = \bigcup_{u \in V} [u]$ である. すなわち, u をそれに接続する辺の本数 (自己ループは 1 個分としてカウント) だけコピーして得られる集合が $[u]$ である.

次に辺集合 E' を以下のように構成する. まず, $(X_n)_{n \in \mathbb{N}}$ を定理 2.2.8 によって構成される n 頂点 d_0-正則 λ-エクスパンダーの族とする. 元のグラフの各頂点 $u \in V$ に対し, $d = |[u]|$ としたとき, 頂点集合 $[u]$ 上で X_d と同型なグラフを構成し, それを $([u], E_u)$ とする. このとき, 各 u-クラウド内部の辺集合 E_{inner} は

$$E_{\text{inner}} = \bigcup_{u \in V} E_u$$

とする. 次に異なるクラウド間を繋ぐ辺集合 E_{outer} を

$$E_{\text{outer}} = \{\{(u, e), (v, e)\} \colon e \notin [u] \cap [v]\}$$

とする. このとき, グラフ G' の辺集合 E' は

$$E' = E_{\text{inner}} \cup E_{\text{outer}}$$

となる.

次に各辺 $e' \in E'$ の制約 $c'_{e'}$ を以下のように定める:

- 辺 $e' \in E_{\text{outer}}$ がクラウド間をつなぐ辺ならば, その制約は対応する元の辺 e の制約と同じとする. すなわち, $e' = \{(u,e),(v,e)\}$ ならば, $c'_{e'} = c_e$ である.
- 辺 $e' \in E_{\text{inner}}$ がクラウド内部の辺ならば, その制約は全集合, すなわち $c'_{e'} = \Sigma^2$ とする.

このようにして得られる制約グラフ $G' = \langle (V', E'), \Sigma, \mathcal{C}' \rangle$ が補題の主張を全て満たすことを確認する. まず, G' は (d_0+1)-正則である. 実際, G' の各頂点 $(u,e) \in V'$ に接続する辺は, クラウド内の辺が d_0 本, クラウド間の辺が 1 本である. また, V' の要素数は次数の総和に等しいため, $|V'| \leq 2|E|$ を満たす ($2|E| - |V'|$ は元のグラフの自己ループの個数分に等しい). 次に, $\mathrm{UNSAT}(G) = 0$ ならば $\mathrm{UNSAT}(G') = 0$ である. 実際, 元の制約グラフの全ての制約を満たす割り当て $a\colon V \to \Sigma$ に対し, G' の割り当て $a'\colon V' \to \Sigma$ を

$$a'(u,e) = a(u)$$

と定めると, a' は G' の制約を満たす (クラウド内の辺に対応する制約は全て満たされ, クラウド間の辺に対応する制約は a の取り方により全て満たされることがわかる).

最後の主張, すなわち $\mathrm{UNSAT}(G') \geq c \cdot \mathrm{UNSAT}(G)$ を背理法で示すために, これが成り立たないと仮定する. このとき, ある G' の割り当て $a'\colon V' \to \Sigma$ が存在して $\mathrm{UNSAT}_{G'}(a') < c \cdot \mathrm{UNSAT}_G(a')$ となる. この割り当て a' に対し, 元のグラフ G の割り当て $a\colon V \to \Sigma$ を

$$a(u) = \mathrm{Maj}((a'(u,e))_{(u,e)\in[u]})$$

と定める. ここで $\mathrm{Maj}(\cdot)$ は多数決関数であり, タイは任意に選ぶとする. 例えば $\mathrm{Maj}(1,2,2) = 2$, $\mathrm{Maj}(1,2,3,3) = 3$, $\mathrm{Maj}(1,2,2,3,3) = 2$ である ($\mathrm{Maj}(1,2,2,3,3) = 3$ としても良いが, ここでは便宜上小さい方の数字を採用している). 以下, このようにして定まる新たな割り当て $a\colon V \to \Sigma$ が $\mathrm{UNSAT}_G(a) < \mathrm{UNSAT}(G')$ となることを示すことによって矛盾を導く (この不等式は $\mathrm{UNSAT}(G)$ の定義に反している).

以降, 割り当て $a'\colon V' \to \Sigma$ に対し, 頂点 $(u,e) \in V'$ への割り当て $a'(u,e)$ を頂点 (u,e) の**意見**と呼ぶこととする. 同様に, 割り当て $a\colon V \to \Sigma$ に対し, 頂点 $u \in V$ への割り当て $a(u)$ を頂点 u の意見と呼ぶこととする. 直感的には頂点 u の意見は対応する u-クラウド内の頂点の意見の多数決である.

辺集合 $F \subseteq E$ を割り当て a によって充足されない辺の集合, すなわち

$$F = \{e = \{u,v\} \in E\colon u < v \text{ and } (a(u), a(v)) \notin c_e\}$$

と定義する. ここで $\mathcal{C} = (c_e)_{e\in E}$ は元の制約グラフ G の制約である. 同様に, $F' \subseteq E'$ を割り当て a' によって充足されない辺の集合とする. このとき $\mathrm{UNSAT}_G(a) = \frac{|F|}{|E|}$ および $\mathrm{UNSAT}_{G'}(a') = \frac{|F'|}{|E'|}$ である. 頂点部分集合 $S \subseteq V'$ を a の多数決で選ばれなかった意見を持つ頂点の集合, すなわち

$$S = \{(u,e) \in V'\colon a(u) \neq a'(u,e)\}$$

と定義し, 各 $v \in V$ に対し $S^v = S \cap [v]$ とし, 各 $\sigma \in \Sigma$ に対し $S^v_\sigma = \{(u,e) \in S^v\colon a'(u,e) = \sigma\}$ とする (図 3.2). なお, $\sigma \in \Sigma$ が多数決の意見 (すなわち $\sigma = a(v)$) のとき, $S^v_\sigma = \emptyset$ とする.

以下の二つのケースを考える:

ケース 1. $|S| \geq \frac{\mathrm{UNSAT}_G(a)}{2}|E|$ の場合. 各頂点 $v \in V$ に対し,

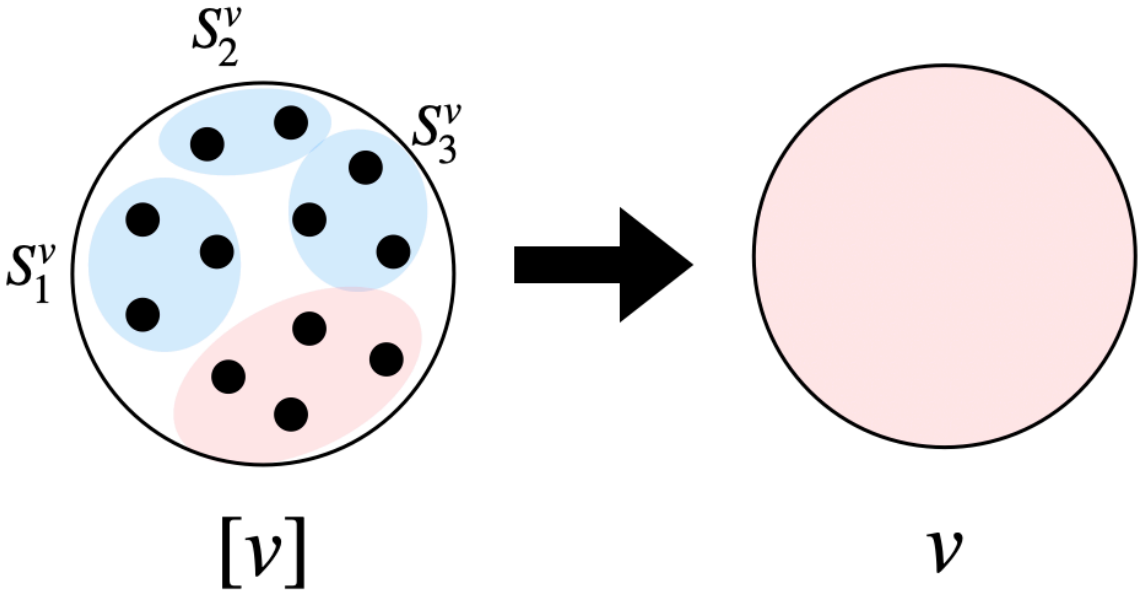

図 3.2: 多数決によって選ばれなかった v-クラウド内の頂点の集合を $S_v \subseteq [v]$ とする.

ケース 2. $|S| < \frac{\mathrm{UNSAT}_G(a)}{2}|E|$ **の場合.** □

3.2.2 エクスパンダー化

3.3 制約グラフのべき乗

3.4 アルファベット削減

Chapter 4

割り当てテスターとアルファベット削減

本チャプターでは, 割り当てテスターとアルファベット削減について述べる.

4.1 割り当てテスター

4.2 アルファベット削減

Bibliography

[AKS04] M. Agrawal, N. Kayal, and N. Saxena. "PRIMES is in P". en. In: **Annals of mathematics** 160 (2 Sept. 1, 2004), pp. 781–793. DOI: 10.4007/annals.2004.160.781 (cit. on p. 11).

[ALMSS98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. "Proof verification and the hardness of approximation problems". In: **Journal of the ACM** 45 (3 May 1, 1998), pp. 501–555. DOI: 10.1145/278298.278306 (cit. on p. 7).

[AS98] S. Arora and S. Safra. "Probabilistic checking of proofs: a new characterization of NP". In: **Journal of the ACM** 45 (1 Jan. 1, 1998), pp. 70–122. DOI: 10.1145/273865.273901 (cit. on p. 7).

[Din07] I. Dinur. "The PCP theorem by gap amplification". In: **Journal of the ACM** 54 (3 June 1, 2007), 12–es. DOI: 10.1145/1236457.1236459 (cit. on p. 7).

[RVW02] O. Reingold, S. Vadhan, and A. Wigderson. "Entropy Waves, the Zig-Zag Graph Product, and New Constant-Degree Expanders". In: **Annals of mathematics** 155 (1 Jan. 2002), p. 157. DOI: 10.2307/3062153 (cit. on p. 22).