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. 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:

- 自然数 $n \in \mathbb{N}$ に対して $[n] = \{1, \ldots, n\}$ とする.
- $\{0,1\}^* = \bigcup_{n\in\mathbb{N}} \{0,1\}^n$ 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.

PCP 定理では主に判定問題と呼ばれる次のような問題を考える:

[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.

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)** という．また, 文字列 $x \in \{0,1\}^*$ 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．

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 1.1.2 （乱択アルゴリズム）

アルゴリズム $A(x;r)$ は, 入力 $x \in \{0,1\}^*$ とは別にランダムシードと呼ばれる別の文字列 $r \in \{0,1\}^*$ を受け取るとき, **乱択アルゴリズム**という．なお, 任意の $x, r \in \{0,1\}^*$ に対して $A(x;r)$ 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 r 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 r の文字数は高々 $T(n)$ であるため, $r \in \{0,1\}^{T(n)}$ であると仮定する．
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,

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

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 1.1.3 （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, $O(\log i)$ 時間を仮定している．

1.2 検証の計算量

このノートでは主に検証の計算量について考える．検証とは, 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, その証明が実際にその命題を正しく証明しているかどうかを判定する問題である．数学の証明の査読業務をイメージしてもらうとわかりやすい

だろう. ただしここで扱う「命題」とは, 一般の数学の命題ではなく, 例えば「特定のグラフ G は 3 彩色を持つ」「線形方程式系 $Ax = b$ は解を持つ」といった, 具体的な問題例を考える.

フォーマルに議論するため, 検証の概念を導入する.

例えば, グラフ G が 3 彩色を持つかどうかという問題は, 3 彩色問題と呼ばれる判定問題であり, 判定問題 $f\colon \{0,1\}^* \to \{0,1\}$ を考える.[2] 検証では**検証者 (verifier)** と呼ばれるアルゴリズム $\mathrm{Ver}(x)$ を考える. このアルゴリズムは入力として

1.2.1 効率的な検証とクラス NP

1.2.2 確率的な検証とクラス MA

1.2.3 局所的な検証とクラス PCP

1.3 PCP 定理

1.3.1 自明な例

定理 1.3.1 （PCP 定理）

…

[2] $\{0,1\}^* = \bigcup_{n\in\mathbb{N}} \{0,1\}^n$ は有限長の二進文字列全体. 判定問題とは, 文字列が与えられたときに Yes または No を答える問題である.

Chapter 2

制約充足問題

制約充足問題 (Constraint Satisfaction Problem, CSP) は, 論理学や計算複雑性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

2.1.1 PCP との関係

2.1.2 制約グラフ

2.2 PCP 定理の証明の概要

2.3 エクスパンダーグラフ

2.3.1 エクスパンダーグラフの定義

2.3.2 エクスパンダー混交補題

2.3.3 制約グラフのエクスパンダー化

Chapter 3

ギャップ増幅補題

この章ではPCP定理の証明において重要な役割を果たすギャップ増幅補題について解説する.

3.1　主張と直感

3.2　証明

3.2.1　構成

3.2.2　正当性の証明

Chapter 4

アルファベット削減

4.1 PCP 定理の証明

Bibliography

[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).