離散数理工学　第 11 回
離散確率論：乱択データ構造とアルゴリズム (発展)

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2021 年 1 月 12 日

最終更新：2021 年 1 月 22 日　17:05

スケジュール 前半

スケジュール 後半 (予定)

注意：予定の変更もありうる

今日の目標

今日の目標

典型的な乱択アルゴリズムの解析ができるようになる
- 多項式の同一性判定 (シュワルツ・ジッペルの補題)
- 多項式の同一性判定の応用：完全マッチングの存在性

重要な技法
- 成功確率の増幅

乱択アルゴリズムの類型

乱択アルゴリズムとは?: 復習

乱数を用いる (あるいは，用いてもよい) アルゴリズムのこと

確率的アルゴリズム，乱数使用アルゴリズムとも呼ばれる

乱択アルゴリズムの類型

乱択アルゴリズムは，主に次の 2 種類に分けられる
- ラスベガス型乱択アルゴリズム
  - 必ず正しい出力を行う
  - 計算量が確率的に定まる
- モンテカルロ型乱択アルゴリズム
  - 正しい出力を行う保証がない
  - 正しい出力に対する「ずれ」が確率的に定まる

目次

多項式の同一性判定 (1)

例題

x に関する次の 2 つの ($\mathbb{R}$ 上の) 多項式は「同じ」多項式か？
- $(x-2)^2-(x+4)(x-3)+(2x-1)^2$
- $4x^2-9x+17$

2 つの多項式 $p, q \in \mathbb{R}[x]$ が同じであるとは，任意の $x \in \mathbb{R}$ に対して

$$p(x) = q(x)$$

となること

多項式の同一性判定 (2)

例題

x に関する次の 2 つの ($\mathbb{R}$ 上の) 多項式は「同じ」多項式か？
- $p(x)=(x-2)^2-(x+4)(x-3)+(2x-1)^2$
- $q(x)=4x^2-9x+17$

展開すれば，次のようになることが分かる

$$\begin{aligned} p(x) &= (x-2)^2-(x+4)(x-3)+(2x-1)^2 \\ &= (x^2-4x+4)-(x^2+x-12)+(4x^2-4x+1) \\ &= (1-1+4)x^2+(-4-1-4)x+(4+12+1) \\ &= 4x^2-9x+17 = q(x) \end{aligned}$$

つまり，上の 2 つの多項式は同じである　□

多項式の同一性判定 (3)

例題

x に関する次の 2 つの ($\mathbb{R}$ 上の) 多項式は「同じ」多項式か？

- $p(x) = (x-2)^2 - (x+4)(x-3) + (2x-1)^2$
- $q(x) = 4x^2 - 9x + 17$

$p(x), q(x)$ は x に関する (高々) 2 次式なので，3 つの異なる点で評価する

- $x = 0$ のとき，$\begin{cases} p(0) = (-2)^2 - 4 \cdot (-3) + (-1)^2 = 17 \\ q(0) = 17 \end{cases}$
- $x = 1$ のとき，$\begin{cases} p(1) = (-1)^2 - 5 \cdot (-2) + 1^2 = 12 \\ q(1) = 4 - 9 + 17 = 12 \end{cases}$
- $x = -1$ のとき，$\begin{cases} p(-1) = (-3)^2 - 3 \cdot (-4) + (-3)^2 = 30 \\ q(-1) = 4 + 9 + 17 = 30 \end{cases}$

つまり，上の 2 つの多項式は同じである □

多項式の同一性判定 (4)

「展開による方法」のよい点と悪い点

- よい点：同じか同じでないか，必ず分かる
- よい点：多変数多項式でも使える
- 悪い点：展開は大変

「評価による方法」のよい点と悪い点

- よい点：同じか同じでないか，必ず分かる
- よい点：多変数多項式でも使える
- よい点：評価は簡単
- 悪い点：多変数多項式だと，評価する点の数が多くなる

多項式の同一性判定 (5)

「評価による方法」のよい点と悪い点

- 悪い点：多変数多項式だと，評価する点の数が多くなる

例えば，

- 1 変数の多項式，次数が高々 d $\rightsquigarrow$ 評価する点の数 $= d + 1$
- 2 変数の多項式，次数が高々 d $\rightsquigarrow$ 評価する点の数 $= \binom{d+2}{2}$

$$p(x, y) = a_{2,0}x^2 + a_{1,1}xy + a_{0,2}y^2 + a_{1,0}x + a_{0,1}y + a_{0,0}$$

- n 変数の多項式，次数が高々 d $\rightsquigarrow$ 評価する点の数 $= \binom{d+n}{n}$

$n = 100$，$d = 10$ $\rightsquigarrow$ $\binom{d+n}{n} = 46,897,636,623,981$ (約 46 兆)

多項式の同一性判定：乱択アルゴリズム

考える乱択アルゴリズム

評価する点をランダムに選ぶ

ランダムに選ぶ方法

- 評価する点の候補となる有限集合 $S \subseteq \mathbb{R}$ を決めておく
- S^n から一様分布に従って点を選ぶ

多項式の同一性判定：乱択アルゴリズム — シュワルツ・ジッペルの補題

考える状況

- $n \geq 1, d \geq 0$： 自然数
- p： n 変数実多項式で，次数が高々 d であり，恒等的に 0 ではないもの (ある $x \in \mathbb{R}^n$ に対して $p(x) \neq 0$)
- $S \subseteq \mathbb{R}$： 有限集合

シュワルツ・ジッペルの補題

S から一様分布に従って独立に $r_1, r_2, \ldots, r_n$ を選ぶとき

$$\Pr(p(r_1, r_2, \ldots, r_n) = 0) \leq \frac{d}{|S|}$$

この補題は Schwartz ('80) と Zippel ('79) によるが，
DeMillo と Lipton ('78) も同じような命題を証明している

シュワルツ・ジッペルの補題を用いた多項式同一性判定

多項式同一性判定に対する乱択アルゴリズム

1. 適当に $S \subseteq \mathbb{R}$ を選ぶ
2. $r_1, \ldots, r_n \in S$ を一様分布に従って独立に選ぶ
3. $p(r_1, \ldots, r_n) - q(r_1, \ldots, r_n) = 0$ ならば，「同一である」と出力
   そうでなければ，「同一でない」と出力

任意の $x \in \mathbb{R}^n$ に対して $p(x) = q(x)$ であるとき，

- 任意の $r_1, \ldots, r_n \in S$ に対して，$p(r_1, \ldots, r_n) = q(r_1, \ldots, r_n)$
- $\therefore$ アルゴリズムは正しく「同一である」と必ず出力

ある $x \in \mathbb{R}^n$ に対して $p(x) \neq q(x)$ であるとき，

- $p - q$ は恒等的に 0 ではない多項式
- シュワルツ・ジッペルの補題から，$p - q$ の次数が d のとき

$$\Pr(p(r_1, \ldots, r_n) - q(r_1, \ldots, r_n) = 0) \leq \frac{d}{|S|}$$

- $\therefore$ 正しく「同一でない」と出力する確率 $\geq 1 - \frac{d}{|S|}$

多項式の同一性判定：よい点と悪い点

「乱択アルゴリズム」のよい点と悪い点

- よい点：評価は簡単
- よい点：多変数多項式でも，評価する点の数が少ない
- 悪い点：同じか同じでないか，必ず分かるわけではない

「展開による方法」のよい点と悪い点

- よい点：同じか同じでないか，必ず分かる
- 悪い点：展開は大変

「評価による方法」のよい点と悪い点

- よい点：同じか同じでないか，必ず分かる
- よい点：評価は簡単
- 悪い点：多変数多項式だと，評価する点の数が多くなる

多項式の同一性判定：よい点と悪い点 (表)

	展開	評価	乱択
必ず分かる	○	○	×
計算が簡単	×	○	○
点の数が少ない	−	×	○

評価する点の数

$n = 100, d = 10$ のとき

- 乱択ではない $\rightsquigarrow \binom{d+n}{n}$ 約 46 兆
- 乱択 $\rightsquigarrow 1$ 1

間違える確率

- 乱択 $\rightsquigarrow$ 同じでないとき，間違える確率 $\leq \frac{d}{|S|}$

間違える確率を小さくするには？ (1)

アイディア 1 : $|S|$ を大きくする

- 例えば，$d = 10$ のとき
  - $|S| = 100$ ならば，$\frac{d}{|S|} = \frac{10}{100} = \frac{1}{10}$
  - $|S| = 200$ ならば，$\frac{d}{|S|} = \frac{10}{200} = \frac{1}{20}$

アイディア 2 : 同じアルゴリズムを繰り返して実行する

間違える確率を小さくするには？ (2)

乱択アルゴリズムの反復実行

K = 反復回数

1. 先ほどの乱択アルゴリズムを K 回独立に実行する
2. 1 回でも $p(r_1, \ldots, r_n) \neq q(r_1, \ldots, r_n) \Rightarrow$ 「同一でない」と出力
   そうでない $\Rightarrow$ 「同一である」と出力

ある $x \in \mathbb{R}^n$ に対して $p(x) \neq q(x)$ であるとき

- K 回とも $p(r_1, \ldots, r_n) = q(r_1, \ldots, r_n)$ となるとき，出力が間違い
- 出力が間違いである確率 $\leq \left(\frac{d}{|S|}\right)^K$

結論：確率増幅

反復実行により，間違う確率を指数関数的に小さくできる

反復実行は，モンテカルロ型乱択アルゴリズムにおける常套手段

目次

シュワルツ・ジッペルの補題 (再掲)

考える状況

- $n \geq 1, d \geq 0$: 自然数
- p : n 変数実多項式で，次数が高々 d であり，恒等的に 0 ではないもの (ある $x \in \mathbb{R}^n$ に対して $p(x) \neq 0$)
- $S \subseteq \mathbb{R}$: 有限集合

シュワルツ・ジッペルの補題 (再掲)

S から一様分布に従って独立に $r_1, r_2, \ldots, r_n$ を選ぶとき

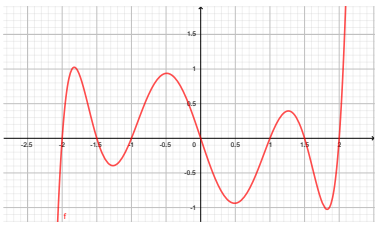
$$\Pr(p(r_1, r_2, \ldots, r_n) = 0) \leq \frac{d}{|S|}$$

証明は n に関する帰納法

シュワルツ・ジッペルの補題：証明 (1)

$n = 1$ のとき

- $p(r) = 0$ となる r の数 $\leq d$ ($\because p$ は高々 d 次多項式)
- $\therefore \Pr(p(r) = 0) \leq \frac{d}{|S|}$



シュワルツ・ジッペルの補題：証明 (2)

$n = k \geq 2$ のとき ($n < k$ のときは成り立つと仮定する)

- $p(x_1, x_2, \ldots, x_k)$ を次の形に書く

$$p(x_1, x_2, \ldots, x_k) = \sum_{i=0}^{d} q_i(x_1, \ldots, x_{k-1}) \cdot x_k^i$$

例 : $k = 3$, $d = 2$

$$p(x_1, x_2, x_3) = \underbrace{(x_1^2 - 3x_1x_2)}_{q_0(x_1,x_2)} + \underbrace{(-2x_1 + x_2 - 4)}_{q_1(x_1,x_2)} x_3 + \underbrace{3}_{q_2(x_1,x_2)} x_3^2$$

- q_i は $n - 1$ 変数多項式で，次数は高々 $d - i$
- ある i に対して，q_i は恒等的に 0 ではない ($\because p$ が恒等的に 0 ではない)
- そのような i の中で最大のものを考え，i^* とする

シュワルツ・ジッペルの補題：証明 (3)

このとき，$p(x_1, x_2, \ldots, x_k) = \sum_{i=0}^{i^*} q_i(x_1, \ldots, x_{k-1}) \cdot x_k^i$

- q_{i^*} は恒等的に 0 ではないので，帰納法の仮定から

$$\Pr(q_{i^*}(r_1, r_2, \ldots, r_{k-1}) = 0) \leq \frac{d - i^*}{|S|}$$

- $q_{i^*}(r_1, r_2, \ldots, r_{k-1}) \neq 0$ という仮定の下で，$p(r_1, r_2, \ldots, r_{k-1}, x_n)$ は 1 変数 i^* 次多項式
- したがって，帰納法の仮定から

$$\Pr(p(r_1, r_2, \ldots, r_{k-1}, r_k) = 0 \mid q_{i^*}(r_1, r_2, \ldots, r_{k-1}) \neq 0) \leq \frac{i^*}{|S|}$$

シュワルツ・ジッペルの補題：証明 (4)

したがって，

$$\begin{aligned}
&\Pr(p(r_1, \ldots, r_k) = 0) \\
&= \Pr(p(r_1, \ldots, r_k) = 0 \mid q_{i^*}(r_1, \ldots, r_{k-1}) = 0) \cdot \Pr(q_{i^*}(r_1, \ldots, r_{k-1}) = 0) \\
&\quad + \Pr(p(r_1, \ldots, r_k) = 0 \mid q_{i^*}(r_1, \ldots, r_{k-1}) \neq 0) \cdot \Pr(q_{i^*}(r_1, \ldots, r_{k-1}) \neq 0) \\
&\leq \Pr(q_{i^*}(r_1, \ldots, r_{k-1}) = 0) + \Pr(p(r_1, \ldots, r_k) = 0 \mid q_{i^*}(r_1, \ldots, r_{k-1}) \neq 0) \\
&\leq \frac{d - i^*}{|S|} + \frac{i^*}{|S|} \\
&= \frac{d}{|S|}
\end{aligned}$$

□

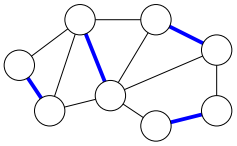
目次

多項式の同一性判定 → 完全マッチングの存在判定

多項式の同一性判定の応用として，
二部グラフにおける完全マッチングの存在判定がある

定義：完全マッチングとは？ (復習)

無向グラフ $G = (V, E)$ の完全マッチングとは，辺部分集合 $M \subseteq E$ で，
各頂点 $v \in V$ に対して，v に接続する M の辺がただ 1 つ存在するもの



そのために，グラフから多変数多項式を構成する

エドモンズ行列

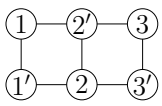
二部グラフ $G = (A, B, E)$，$|A| = |B|$，
$A = \{1, 2, \ldots, n\}, B = \{1', 2', \ldots, n'\}$

定義：エドモンズ行列

G のエドモンズ行列とは，次の $n \times n$ 行列 $M_G = (m_{i,j})$

$$m_{i,j} = \begin{cases} x_{i,j} & (\{i, j'\} \in E \text{ のとき}) \\ 0 & \text{それ以外のとき} \end{cases}$$

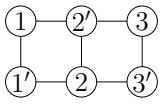
ただし，辺 $\{i, j'\}$ に対して，$x_{i,j}$ は変数

$G =$ 

$$M_G = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1' & 2' & 3' \end{array} \\ \begin{pmatrix} x_{1,1} & x_{1,2} & 0 \\ x_{2,1} & x_{2,2} & x_{2,3} \\ 0 & x_{3,2} & x_{3,3} \end{pmatrix} \end{array}$$

エドモンズ行列の行列式

$$\det(M_G) = \begin{vmatrix} x_{1,1} & x_{1,2} & 0 \\ x_{2,1} & x_{2,2} & x_{2,3} \\ 0 & x_{3,2} & x_{3,3} \end{vmatrix}$$
$$= x_{1,1}x_{2,2}x_{3,3} - x_{1,1}x_{2,3}x_{3,2} - x_{1,2}x_{2,1}x_{3,3}$$

$G =$ 

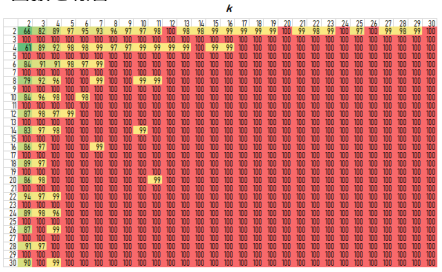
- 行列式は $(x_e \mid e \in E)$ に関する多項式　(変数の数 $= |E|$，次数 $= n$)
- $\det(M_G)$ の各単項式が G の完全マッチングに対応する
  (c.f. 第 5 回講義)

$\therefore \det(M_G)$ が恒等的に 0 である $\Leftrightarrow$ G に完全マッチングが存在しない

実験

G が偶数長の閉路であるとき，$\det(M_G)$ が恒等的に 0 か，
乱択アルゴリズムで調べた

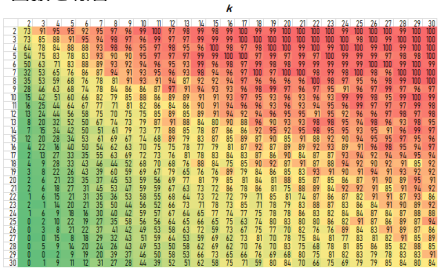
- $n \in \{2, 3, \ldots, 30\}, S = \{1, 2, \ldots, k\}, k \in \{2, 3, \ldots, 30\}$
- Ruby の rand で乱数生成
- 乱択アルゴリズムを 100 回実行し，「恒等的に 0 ではない」と
  出力した回数を報告



実験 (続き)

G が偶数長の閉路であるとき，$\det(M_G)$ が恒等的に 0 か，
乱択アルゴリズムで調べた

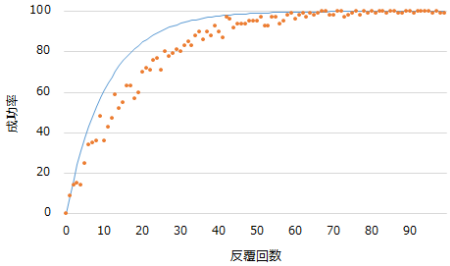
- $n \in \{2, 3, \ldots, 30\}, S = \{-k, \ldots, k\}, k \in \{2, 3, \ldots, 30\}$
- Ruby の rand で乱数生成
- 乱択アルゴリズムを 100 回実行し，「恒等的に 0 ではない」と
  出力した回数を報告



実験 (続き 2)

乱択アルゴリズムを反復実行し，成功率増幅の様子を観察した

- $n = 30, S = \{-k, \ldots, k\}, k = 4$，Ruby の rand で乱数生成
- 乱択アルゴリズムの反復回数 $\in \{0, 1, \ldots, 100\}$
- 100 回実行し，正しく報告した回数を報告



目次

今日の目標

今日の目標

典型的な乱択アルゴリズムの解析ができるようになる

- 多項式の同一性判定 (シュワルツ・ジッペルの補題)
- 多項式の同一性判定の応用：完全マッチングの存在性

重要な技法

- 成功確率の増幅