計算機数学I (2019)
第15回

照井 章（筑波大学 数理物質系 数学域）
Akira Terui (Institute of Mathematics, University of Tsukuba)

第14回のまとめ

- 高速乗算法:

  Karatsuba（カラツバ）の乗算アルゴリズム

  - 整数の乗算アルゴリズム
  - 1変数多項式の乗算アルゴリズム

第15回の内容

- 高速乗算法：高速フーリエ変換 (FFT)
  - 離散フーリエ変換 (DFT)
  - 高速フーリエ変換 (FFT)
  - FFTを用いた1変数多項式の高速乗算法

多項式の評価と補間 (§ 6.1) (p.85)

多項式の表現

- 係数表現

  $f(x) = a_n x^n + \cdots + a_1 x + a_0 \in \mathrm{R}[x]$

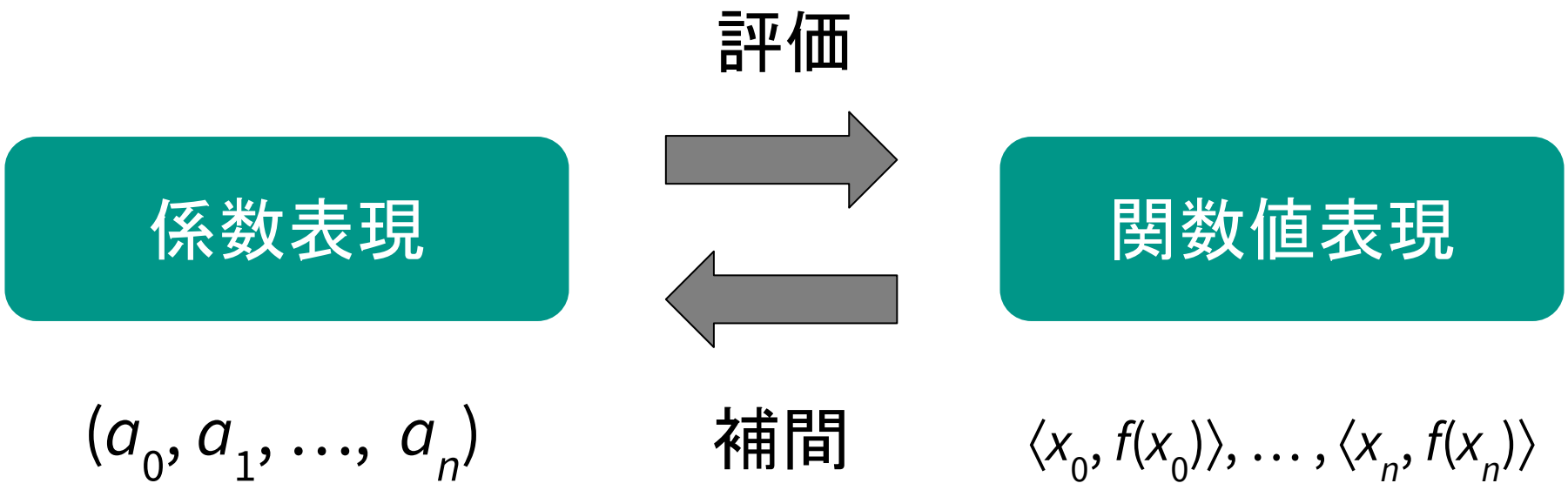
  $\leftrightarrow (a_0, a_1, \ldots, a_n)$

- 関数値表現

  $\langle x_0, f(x_0)\rangle, \ldots, \langle x_n, f(x_n)\rangle$: n+1点の値で表す

どちらの表現もn次以下の多項式を一意に表す

多項式の評価と補間


評価
係数表現
関数値表現
$(a_0, a_1, \ldots, a_n)$
補間
$\langle x_0, f(x_0)\rangle, \ldots, \langle x_n, f(x_n)\rangle$

多項式の補間の一意性（定理 6.1）

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$$

Vandermonde行列
(x_iが互いに異なれば正則)

係数表現

関数値表現

多項式の古典的な評価や補間の計算量は O(n^2)

- 評価: Horner法

  $\langle x_0, f(x_0)\rangle, \ldots, \langle x_n, f(x_n)\rangle$

  計算量: (n+1) O(n) = O(n^2)

- 補間: (例) Lagrangeの補間公式（定理 6.2）

$$a(x) = \sum_{k=0}^{n} a(x_k) \prod_{j \neq k} \frac{x - x_j}{x_k - x_j}$$

  計算量: O(n^2)

高速な評価や補間のアルゴリズムと多項式の加算／乗算への応用

評価: $O(n \lg n)$

補間: $O(n \lg n)$

をみたすアルゴリズムが存在する

関数値表現による多項式の加算／乗算の効率化

$f(x) \leftrightarrow \langle x_0, f(x_0)\rangle, \ldots, \langle x_n, f(x_n)\rangle$

$g(x) \leftrightarrow \langle x_0, g(x_0)\rangle, \ldots, \langle x_n, g(x_n)\rangle$

のとき

$f(x)+g(x) \leftrightarrow \langle x_0, f(x_0)+g(x_0)\rangle, \ldots, \langle x_n, f(x_n)+g(x_n)\rangle$

$f(x)g(x) \leftrightarrow \langle x_0, f(x_0)g(x_0)\rangle, \ldots, \langle x_n, f(x_n)g(x_n)\rangle$

を計算時間 $O(n)$ で計算可能

たたみこみ積
(convolution)

関数値表現による多項式の加算／乗算の効率化

合計 $O(n \lg n)$ の計算量で多項式乗算を行う

[Step 1] $f(x), g(x)$ の多項式表現を関数値表現に変換: $O(n \lg n)$ 離散フーリエ変換

[Step 2] $f(x) * g(x)$（たたみこみ積）を計算: $O(n)$

[Step 3] $f(x)g(x)$ の関数値表現を多項式表現に変換: $O(n \lg n)$ 離散フーリエ逆変換

高速フーリエ変換 (FFT) で $O(n \lg n)$ の変換

高速フーリエ変換
(Fast Fourier Transformation: FFT) (§ 6.2)

1の原始n乗根

- 定義（1の原始n乗根）

  K: 体

  $\omega \in K$ が1の原始n乗根 $\Leftrightarrow \omega^n = 1, \omega^k \neq 1$ for $0<k<n$

- $K = \mathbf{C}$, $n > 0$のとき:

  1の原始 n 乗根: $\omega_n = \exp(2\pi i / n)$

  1の n 乗根: $\omega_n^k = \exp(2\pi k i / n), k \in Z$

1のn乗根の性質

- $n \geqq 0, k \geqq 0$ のとき$(\omega_{dn})^{dk} = (\omega_n)^k$
  例: mが偶数のとき $(\omega_m)^{m/2} = \omega_2 = -1$
  ($d = m/2, n = 2, k = 1$)
- $n > 0$ が偶数 ⇒ (1のn乗根)^2 = (1の$n/2$乗根)
  (n個) ($n/2$個)
- (総和補題) n: 正整数、$k \in \mathbb{Z}$ に対し

$$\sum_{j=0}^{n-1} \omega_n^{kj} = \begin{cases} 0 & (k \nmid n) \\ n & (k \mid n) \end{cases}$$

離散フーリエ変換
(Discrete Fourier Transform: DFT)

$a(x)$: $\mathbf{C} \to \mathbf{C}$ のとき

(ω_n に関する) f の離散フーリエ変換
(Discrete Fourier Transform: DFT)

$\Leftrightarrow (a(1), a(\omega_n), \ldots, a(\omega_n^{n-1}))$ を求めること

高速フーリエ変換
（Fast Fourier Transform: FFT）

$n = 2^k$ とする

$$a(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$$

$$= (a_{n-1}x^{n-1} + \cdots + a_3 x^2 + a_1)\, x + (a_{n-2}x^{n-2} + \cdots + a_2 x^2 + a_0)$$

高速フーリエ変換
(Fast Fourier Transform: FFT)

$$b(x) = (a_{n-2}x^{n/2-1} + \cdots + a_2 x + a_0) \text{ (偶数次の項),}$$

$$c(x) = (a_{n-1}x^{n/2-1} + \cdots + a_3 x^1 + a_1) \quad \text{(奇数次の項)}$$

とおくと

$$a(x) = b(x^2) + x\,c(x^2)$$

高速フーリエ変換
(Fast Fourier Transform: FFT)

このとき

$$a(\omega_n^{\,k}) = b(\omega_{n/2}^{\,k}) + (\omega_n^{\,k})\, c(\omega_{n/2}^{\,k})$$

再帰的に計算　　再帰的に計算

$$a(\omega_n^{\,n/2+k}) = b(\omega_{n/2}^{\,k}) - (\omega_n^{\,k})\, c(\omega_{n/2}^{\,k})$$

下半平面の計算には上半平面の結果が使える

FFTのアルゴリズム

Algorithm FFT $(e, \omega_n, a(x))$

入力:

- $e \in \mathbb{N}$ s.t $n = 2^e$
- ω_n : 1の原始n乗根
- $a(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 = (a_0, a_1, \ldots, a_{n-1})$

出力:

$a(x)$ の ω_n に関する DFT $\{a(1), a(\omega_n), \ldots, a(\omega_n^{n-1})\}$

FFTのアルゴリズム

1. if $e = 0$ then return a;
2. $b(\mathrm{x}) \leftarrow (a_0, a_2, \ldots, a_{n-2})$;
3. $c(\mathrm{x}) \leftarrow (a_1, a_3, \ldots, a_{n-1})$;
4. $y \leftarrow \mathrm{FFT}(e-1, \omega_{n/2}, b(\mathrm{x}))$;
5. $z \leftarrow \mathrm{FFT}(e-1, \omega_{n/2}, c(\mathrm{x}))$;
6. $\mathrm{u} \leftarrow 1$;

FFTのアルゴリズム

7. for i $\in$ [0 .. $n/2 - 1$] do
   a. $x_i \leftarrow y_i + u\, z_i$;
   b. $x_{n/2+i} \leftarrow y_i - u\, z_i$;
   c. $u \leftarrow \omega_n \cdot u$;
8. return $(x_0, x_1, \ldots, x_{n-1})$;

FFTの計算例

$A(\mathrm{x}) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 = (a_0, a_1, a_2, a_3),\ \omega_4 = \exp(\pi\,\mathrm{i} / 2)$

に対し

$A(\omega_4{}^0) = A(1) = a_0 + a_1 + a_2 + a_3$

$A(\omega_4{}^1) = A(\mathrm{i}) = a_0 + a_1\,\mathrm{i} - a_2 - a_3\,\mathrm{i}$

$A(\omega_4{}^2) = A(-1) = a_0 - a_1 + a_2 - a_3$

$A(\omega_4{}^3) = A(-\mathrm{i}) = a_0 - a_1\,\mathrm{i} - a_2 + a_3\,\mathrm{i}$

を求める

FFTの計算例

(1) FFT $(2, \omega_4, A(x))$ の計算:

$$A(x) = (a_0 + a_2 x^2) + (a_1 + a_3 x^2)\, x = A_0(x) + A_1(x)\, x$$

$$y = r_0 = \mathrm{FFT}(1, \omega_2, A_0(x^2)); \rightarrow (2)$$

$$z = r_1 = \mathrm{FFT}(1, \omega_2, A_1(x^2)); \rightarrow (3)$$

FFTの計算例

(2) $\mathrm{FFT}(1, \omega_2, A_0(x^2))$の計算:

$$A_0(x^2) = a_0 + a_2 x = A_{0,0}(x) + A_{0,1}(x)\, x$$

$$r_{0,0} = \mathrm{FFT}(0, \omega_1, A_{0,0}(x^2)) = a_0$$

$$r_{0,1} = \mathrm{FFT}(0, \omega_1, A_{0,1}(x^2)) = a_2$$

よって

$$(x_0 = a_0 + a_2, x_1 = a_0 - a_2)$$

FFTの計算例

(3) $\mathrm{FFT}(1, \omega_2, A_1(x^2))$の計算:

$$A_1(x^2) = a_1 + a_3 x = A_{1,0}(x) + A_{1,1}(x)\, x$$

$$r_{1,0} = \mathrm{FFT}(0, \omega_1, A_{1,0}(x^2)) = a_1$$

$$r_{1,1} = \mathrm{FFT}(0, \omega_1, A_{1,1}(x^2)) = a_3$$

よって

$$(x_0 = a_1 + a_3, x_1 = a_1 - a_3)$$

FFTの計算例

(1) FFT $(2, \omega_4, A(x))$ の計算（続き）

$$A(x) = (a_0 + a_2 x^2) + (a_1 + a_3 x^2)\, x = A_0(x) + A_1(x)\, x$$

$$y = \mathrm{FFT}(1, \omega_2, A_0(x^2)) = (a_0 + a_2,\ a_0 - a_2)$$

$$z = \mathrm{FFT}(1, \omega_2, A_1(x^2)) = (a_1 + a_3,\ a_1 - a_3)$$

$$r = 1$$

FFTの計算例

(1) FFT $(2, \omega_4, A(x))$ の計算（続き）

$$x_0 \leftarrow y_0 + z_0 = a_0 + a_2 + a_1 + a_3 = A(\omega_4{}^0)$$

$$x_2 \leftarrow y_0 - z_0 = a_0 + a_2 - a_1 - a_3 = A(\omega_4{}^2)$$

$$u \leftarrow \omega_4 \cdot u = \omega_4 = \mathrm{i}$$

$$x_1 \leftarrow y_1 + \omega_4 z_1 = a_0 - a_2 + \mathrm{i}\,(a_1 - a_3) = A(\omega_4{}^1)$$

$$x_2 \leftarrow y_1 - \omega_4 z_1 = a_0 - a_2 - \mathrm{i}\,(a_1 - a_3) = A(\omega_4{}^3)$$

離散逆フーリエ変換 (Inverse DFT, IDFT)

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \cdots & \omega_n^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & \omega_n^{n-1} & \cdots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f(1) \\ f(\omega_n) \\ \vdots \\ f(\omega_n^{n-1}) \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix}$$

V: Vandermonde行列

係数表現

関数値表現

とおくと、上の式はDFTを表す

離散逆フーリエ変換 (Inverse DFT, IDFT)

もし V^{-1} が見つかれば、逆変換は

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = V^{-1} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix}$$

で表される

離散逆フーリエ変換 (Inverse DFT, IDFT)

ところが

$$W = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega_n^{-1} & \cdots & \omega_n^{-n+1} \\ \vdots & \vdots & & \vdots \\ 1 & \omega_n^{-n+1} & \cdots & \omega_n^{(-n+1)(-n+1)} \end{pmatrix}$$

とおくと、総和補題より $W V = n E_n$

ゆえに $V^{-1} = (1/n) W$

離散逆フーリエ変換 (Inverse DFT, IDFT)

さらに、ω_n が1の原始n乗根であることから

$\omega_n^{-1} = \omega_n^{n-1}$ もまた1の原始n乗根

よって IDFT は $(1/n)\,\mathrm{FFT}(e, \omega_n^{-1}, b(x))$ で計算可能

ここで $b(x) = b_{n-1}x^{n-1} + \cdots + b_1 x + b_0$

FFTの計算量評価

FFTの計算量

定理 ω_n が1の原始n乗根のとき

次数 $n = 2^e$ 未満の多項式に対するFFTの計算量は

$O(n \lg n)$.

証明:

$T(n)$: n 次未満の多項式 f に対するFFTの計算量

$T(1) := 0$

とおく

FFTの計算量

1. if $e = 0$ then return a;
2. $b(\mathrm{x}) \leftarrow (a_0, a_2, \ldots, a_{n-2})$;
3. $c(\mathrm{x}) \leftarrow (a_1, a_3, \ldots, a_{n-1})$;
4. $y \leftarrow \mathrm{FFT}(\mathrm{e}-1, \omega_{n/2}, b(\mathrm{x}))$;
5. $z \leftarrow \mathrm{FFT}(\mathrm{e}-1, \omega_{n/2}, c(\mathrm{x}))$;
6. $\mathrm{u} \leftarrow 1$;

T(n/2)

T(n/2)

FFTの計算量

7. for i ∈ [0 .. $n/2 - 1$] do
   a. $x_i \leftarrow y_i + u\, z_i$;
   b. $x_{n/2+i} \leftarrow y_i - u\, z_i$;
   c. $u \leftarrow \omega_n \cdot u$;
8. return $(x_0, x_1, \ldots, x_{n-1})$;

ループ $n/2$ 回

加算1回、乗算1回

加算1回、乗算1回

乗算1回

ゆえに $T(n) = 2 \cdot T(n/2) + 5\,(n/2) = 2 \cdot T(n/2) + O(n/2)$

FFTの計算量

$n = 2^e$ を代入して整理すると

$$T(2^e) = 2 \cdot T(2^{e-1}) + O(2^{e-1})$$

$$= 2 \cdot (2 \cdot T(2^{e-1}) + O(2^{e-1})) + O(2^{e-1})$$

$$= 2^3 T(2^{e-1}) + 3\, O(2^{e-1})$$

$$= \cdots$$

$$= 2^e T(1) + e\, O(2^{e-1}) = O(e \cdot 2^e) = O(n \lg n)$$

FFTを用いた1変数多項式の乗算アルゴリズム

FFTを用いた1変数多項式の乗算アルゴリズム

Algorithm FFT-multiply $(e, a(x), b(x))$

入力:

- $e \in \mathrm{N}$ s.t $n = 2^e$
- $a(x) = (a_0, a_1, \ldots, a_{n-1}) \in \mathbf{C}[x]$
- $b(x) = (b_0, b_1, \ldots, b_{n-1}) \in \mathbf{C}[x]$

出力: $c(x) = (c_0, c_1, \ldots, c_{2n-1}) \in \mathbf{C}[x]$

FFTを用いた1変数多項式の乗算アルゴリズム

1. $(x_0, x_1, \ldots, x_{2n-1}) \leftarrow \mathrm{FFT}(e+1, \omega_{2n}, a(x))$;
2. $(y_0, y_1, \ldots, y_{2n-1}) \leftarrow \mathrm{FFT}(e+1, \omega_{2n}, b(x))$;
3. for $i \in [0 .. 2n - 1]$ do
   a. $z_i \leftarrow x_i y_i$;
4. $(c_0, c_1, \ldots, c_{2n-1}) \leftarrow (1/n)\,\mathrm{FFT}(e+1, (\omega_{2n})^{-1}, (z_0, z_1, \ldots, z_{2n-1}))$;
5. return $(c_0, c_1, \ldots, c_{2n-1})$;

FFT乗算アルゴリズムの計算量

1. $(x_0, x_1, \ldots, x_{2n-1}) \leftarrow \mathrm{FFT}(e+1, \omega_{2n}, a(x))$;
2. $(y_0, y_1, \ldots, y_{2n-1}) \leftarrow \mathrm{FFT}(e+1, \omega_{2n}, b(x))$;
3. for $i \in [0 \,..\, 2n-1]$ do
   a. $z_i \leftarrow x_i y_i$;
4. $(c_0, c_1, \ldots, c_{2n-1}) \leftarrow (1/n)\,\mathrm{FFT}(e+1, (\omega_{2n})^{-1}, (z_0, z_1, \ldots, z_{2n-1}))$;
5. return $(c_0, c_1, \ldots, c_{2n-1})$;

O(n lg n)

O(n lg n)

O(n)

O(n lg n)

合計 O(n lg n)

第15回のまとめ

- 高速乗算法：高速フーリエ変換 (FFT)
  - 離散フーリエ変換
  - 高速フーリエ変換
  - FFTを用いた1変数多項式の高速乗算法