Быстрое преобразование Фурье

Косолобов А.
11.02.2026
Техал - Кирилл **Ипотека**

Инволюция и умножение многочленов

Определение 1. Даны последовательности чисел $a_0, \dots, a_{n-1}$ и $b_0, \dots, b_{n-1}$. Последовательность $c_0, \dots, c_{2n-2}$ задаётся формулой

$$c_k = \sum_{t=0}^{k} a_t b_{k-t}.$$

(считаем $a_i = b_i = 0$ при $i \notin [0, n-1]$). Эта операция называется **инволюцией (сверткой)**:

$$c = a * b.$$

Умножение многочленов

Рассмотрим многочлены:

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

$$B(x) = b_0 + b_1 x + \cdots + b_{n-1} x^{n-1}.$$

Тогда их произведение:

$$C(x) = A(x)B(x) = c_0 + c_1 x + \cdots + c_{2n-2} x^{2n-2},$$

где коэффициенты c_k задаются инволюцией.

Сложность алгоритмов

- Наивное умножение: $O(n^2)$.
- Алгоритм Карацубы:
$$O\left(n^{\log_2 3}\right).$$
- Алгоритм с быстрым преобразованием Фурье (FFT):
$$O(n \log n).$$

Идея алгоритма

План умножения многочленов:

1. Выберем $2n-1$ точек:
$$\omega_0, \omega_1, \ldots, \omega_{2n-2}.$$
2. Посчитаем значения:
$$A(\omega_k), \quad B(\omega_k).$$
3. Перемножим:
$$y_k = A(\omega_k)B(\omega_k).$$
4. Восстановим многочлен $C(x)$ по его значениям в $2n-1$ точках (интерполяция).

Наивно вычисление значений и интерполяция дают сложность $O(n^2)$.

FFT позволяет выбрать точки ω_k так, что шаги вычисления значений и обратного восстановления выполняются за
$$O(n \log n).$$

Реализация

Достаточно уметь находить быстро значения многочлена в точках $\omega_0 \ldots \omega_{n-1}$ и наоборот: по точкам находить коэффициенты. Считаем, что $n = 2^t$, иначе добьем нулями.

Из коэффициентов получаем значения (Прямое FFT)

Алгоритм выбирает $\omega_0 \ldots \omega_{n-1}$:

$$\omega_k = cos\frac{2\pi k}{n} + isin\frac{2\pi k}{n} = e^{i\frac{2\pi k}{n}}$$

- комплексные корни из единицы.

$$\omega_k = w_1^k = w^k$$

Разделим A по четным и нечетным коэффициентам. Пусть:

$$A_0(x) = a_0 + a_2 x + \ldots a_{n-2} x^{\frac{n-2}{2}}$$

$$A_1(x) = a_1 + a_3 x + \ldots a_{n-1} x^{\frac{n-2}{2}}$$

Тогда:
$$A(x) = A_0(x^2) + xA_1(x^2)$$

Посчитаем значение в точке ω_k, используя свойства корней из единицы:

$$y_k = A(\omega_k) = A(\omega^k) = A_0(\omega^{2k}) + \omega^k A_1(\omega^{2k})$$

Также, для удобства введем обозначение:

$$y_k^{(0)} = A_0(\omega^{2k});\ y_k^{(1)} = A_1(\omega^{2k})$$

Т.к. $\omega^{2k} = \omega^{2k+n}$, то нам достаточно вычислить значения $y_k^{(0)}$ и $y_k^{(1)}$ от нуля до $\frac{n}{2} - 1$. Вычислим рекурсивно значение многочлена, а затем:

$$y_k = y_k^{(0)} + \omega_k y_k^{(1)}$$

$$y_{\frac{n}{2}+k} = y_k^{(0)} + \omega_{\frac{n}{2}+k} y_k^{(1)}$$

Оценка времени работы прямого FFT

На каждом шаге рекурсии степень многочлена уменьшается вдвое, как и количество точек для вычсления, количество подзадач - удваивается, после выполнения рекурсии идет подсчет значений за линию, следовательно время работы можно оценить, как:

$$\mathcal{T}(n) = 2\mathcal{T}\left(\frac{n}{2}\right) + O(n) = O(n \log n)$$

Из значений получаем коэффициенты (Обратное FFT)

Матричное представление прямого FFT:

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_1 & \omega_1^2 & \cdots & \omega_1^{n-1} \\ 1 & \omega_2 & \omega_2^2 & \cdots & \omega_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_{n-1} & \omega_{n-1}^2 & \cdots & \omega_{n-1}^{n-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

F_ω - **матрица Вандермонда** для $\omega_1 \ldots w_{n-1}$

Утверждение 1.

$$F_{\omega^{-1}} \cdot F_\omega = n \cdot \mathbf{I}$$

Где I - единичная матрица.

□

$$\forall\ i, j\ \ \in [0...n-1]\ \sum_{k=0}^{n-1} (\omega^i)^k \cdot (\omega^{-j})^k = \sum_{k=0}^{n-1} (\omega^{i-j})^k$$

Используя свойство арифмитической прогрессии получаем, что сумма равна:

$$\frac{(\omega^{i-j})^n - 1}{\omega^{i-j} - 1} = \begin{cases} 0,\ i \neq j \\ n, i = j \end{cases}$$

■

Получаем, что обратное преобразование Фурье (значение коэффициентов по значениям в точках) - это прямое FFT с ω^{-1}, деленное на n.

Оценка времени работы всего алгоритма

Т.к. весь алгоритм это по сути: три раза применить FFT, то его сложность также останется $O(n\ log\ n)$.

Обобщение FFT (Number-Theoretic Transform)

Вместо $\mathbb{C}$ можно использовать область целостности с единицей, в которой есть циклическая подгруппа по умножению порядка n и элемент $1+1+\dots+1$ (n раз единица) должен иметь обратный, т.к. в реализации на $\mathbb{C}$ мы делим на n.

Пример

$R = \mathbb{Z}_p \;\; p \in \mathbb{P}$ и

$$n = 2^k$$

$$p = c \cdot 2^k + 1$$

Тогда $\exists\ g \in \mathbb{Z}_p$ такой, что он пораждает группу порядка $p-1$. А за аргумент возьмем:

$$\omega = g^c \mod p$$

Приложение

Предположим, нам даны значения сигнала в точках $0, 1, \dots n-1$: $a_0, a_1, \dots a_{n-1}$. Хотим узнать насколько эта последовательность похожа на:

$$sin(\frac{2\pi k}{n}x)\ ;\ cos(\frac{2\pi k}{n}x)$$

Тогда пусть:

$$Im(y_k) = \sum_{t=0}^{n-1} a_t sin(\frac{2\pi k}{n}t)$$

$$Re(y_k) = \sum_{t=0}^{n-1} a_t cos(\frac{2\pi k}{n}t)$$

$$w_k^t = cos(\frac{2\pi k}{n}t) + isin(\frac{2\pi k}{n}t)$$

$$\sum_{t=0}^{n-1} a_t w_k^t = y_k$$

Смысл коэффициентов

Полученная величина y_k показывает, насколько сигнал $\{a_t\}$ содержит колебание частоты k. Действительно, числа

$$\sum_{t=0}^{n-1} a_t \cos\left(\frac{2\pi k}{n} t\right) \quad \text{и} \quad \sum_{t=0}^{n-1} a_t \sin\left(\frac{2\pi k}{n} t\right)$$

- это скалярные произведения сигнала с косинусом и синусом данной частоты.

Если сигнал похож на колебание частоты k, то значения a_t будут согласованы по фазе с соответствующими синусом и косинусом, и сумма окажется большой по модулю.

Если же сигнал не содержит такой частоты, то положительные и отрицательные вклады будут взаимно компенсироваться, и сумма будет близка к нулю.

Таким образом, модуль комплексного числа

$$|y_k|$$

показывает степень присутствия частоты k в сигнале.

Чем больше $|y_k|$, тем сильнее в сигнале выражена гармоника с частотой k.

Иными словами, дискретное преобразование Фурье раскладывает сигнал по базису синусов и косинусов и измеряет вклад каждой частоты.