

# Лекция 2 от 06.09.2016. Быстрое преобразование Фурье

Чтобы быть успешным программистом, надо знать 3 вещи:

- Сортировки;
- Хэширование;
- Преобразование Фурье.

---

Глеб

В этой лекции будет разобрано дискретное преобразование Фурье (Discrete Fourier Transform).

## Применение преобразования Фурье

Допустим, что мы хотим решить такую задачу:

**Пример 1.** Даны 2 бинарные строки  $A$  и  $B$  длины  $n$  и  $m$  соответственно. Мы хотим найти, какая подстрока в  $A$  наиболее похожа на  $B$ . Наивная реализация решает эту задачу в худшем случае за  $O(n^2)$ . Преобразование Фурье поможет решить эту задачу за  $O(n \log n)$ , а именно научимся решать другую задачу:

**Цель.** Хотим научиться перемножать многочлены одной степени

$A(x) = a_0 + a_1x + \dots + a_{n-1}x^{n-1}$  и  $B(x) = b_0 + b_1x + \dots + b_{n-1}x^{n-1}$  так, что  $C(x) = A(x)B(x)$ , то есть считать свёртку (найти все коэффициенты, если по-другому)  $\sum_{i=0}^{n-1} \sum_{j=0}^i a_j b_{i-j} x^i$  за  $O(n \log n)$ .

Вернёмся к нашему примеру. Поймём как с помощью нашей **цели** решать задачу про бинарные строки.

Пусть  $A = a_0 \dots a_{n-1}$ ,  $B = b_0 \dots b_{n-1}$ . Их можно считать одной длины (просто добавим нулей в конец  $b$  при надобности). Теперь задача переформулировывается как нахождение максимального скалярного произведения  $B$  и некоторых циклических сдвигов  $A$  (до  $n - m + 1$ ).

Инвертируем массив  $B$  и припишем в конец  $n$  нулей, а к массиву  $A$  припишем самого себя. Посмотрим на все коэффициенты перемножения:

$$c_k = \sum_{i+j=k} a_i b_j$$

Но  $b_i = 0$  при  $i \geq n$ , поэтому при  $k \geq n$ :

$$c_k = \sum_{i=0}^{n-1} b_i a_{k-i}$$

Выбрав нужные коэффициенты, мы решили эту задачу.

## Алгоритм быстрого преобразования Фурье

Основная идея алгоритма заключается в том, чтобы представить каждый многочлен через набор  $n$  точек и значений многочлена в этих точках, быстро (за  $O(n \log n)$ ) вычислить значения в каких-то  $n$  точках для обоих многочленов, потом перемножить за  $O(n)$  сами значения. Потом применить обратное преобразование Фурье и получить коэффициенты  $C(x) = A(x)B(x)$ .

Итак, для начала будем считать, что  $n = 2^k$  (просто добавим нулей до степени двойки).

Рассмотрим циклическую группу корней из 1 —  $W_n = \{e^{i\frac{2\pi k}{n}} \mid k = 0, \dots, n-1\}$ . Обозначим за  $w_n = e^{i\frac{2\pi}{n}}$ . Одно из самых главных свойств, что  $w_n^p \cdot w_n^q = w_n^{p+q}$ , которым мы будем пользоваться в дальнейшем.

Воспользуемся идеей метода «разделяй и властвуй».

Пусть  $A(x) = a_0 + \dots + a_{n-1}x^{n-1}$ .

Представим  $A(x) = A_l(x^2) + xA_r(x^2)$  так, что

$$A_l(x^2) = a_0 + a_2x^2 + \dots + a_{n-2}x^{n-2}$$

$$A_r(x^2) = a_1 + a_3x^2 + \dots + a_{n-1}x^{n-2}$$

**Определение 1.** Назовём *Фурье-образом* многочлена  $P(x) = p_0 + \dots + p_{m-1}x^{m-1}$  вектор из  $m$  элементов —  $\langle P(1), P(w_m), P(w_m^2), \dots, P(w_m^{m-1}) \rangle$ .

Теперь рекурсивно запускаемся от многочленов меньшей степени. Так как для любого целого неотрицательного  $k$  следует, что  $2k$  четное число, то  $w_n^{2k} = w_{n/2}^k \in W_{n/2}$ , то есть мы можем уже использовать значения Фурье-образа для вычисления  $A(x)$ .

Если мы сможем за линейное время вычислить сумму  $A_l(x^2) + xA_r(x^2)$ , то суммарное время работы будет  $O(n \log n)$ , так как  $A_l(x), A_r(x)$  имеют степень в 2 раза меньше, чем  $A(x)$ .

Действительно это легко сделать из псевдокода, который приведен ниже:

---

### Algorithm 1 FFT

---

```

1: function FFT( $A$ )  $\triangleright$   $A$  — массив из комплексных чисел, функция возвращает Фурье-образ
2:    $n \leftarrow \text{length}(A)$ 
3:   if  $n == 1$  then
4:     return  $A$ 
5:    $A_l \leftarrow \langle a_0, a_2, \dots, a_{n-2} \rangle$ 
6:    $A_r \leftarrow \langle a_1, a_3, \dots, a_{n-1} \rangle$ 
7:    $\hat{A}_l \leftarrow \text{FFT}(A_l)$ 
8:    $\hat{A}_r \leftarrow \text{FFT}(A_r)$ 
9:   for  $k \leftarrow 0$  to  $\frac{n}{2} - 1$  do
10:     $A[k] \leftarrow \hat{A}_l[k] + e^{i\frac{2\pi k}{n}} \hat{A}_r[k]$ 
11:     $A[k + \frac{n}{2}] \leftarrow \hat{A}_l[k] - e^{i\frac{2\pi k}{n}} \hat{A}_r[k]$   $\triangleright$  Здесь минус перед комплексным числом из-за того,
    что мы должны найти другой угол, удвоенный которого на окружности будет  $\frac{2\pi(k+n/2)}{n}$ 
12:   return  $A$ 
```

---

Теперь поговорим про обратное FFT. Этого материала не было на лекции на момент написания:

Фактически, мы вычислили такую вещь за  $O(n \log n)$ :

$$\begin{pmatrix} w_n^0 & w_n^0 & w_n^0 & w_n^0 & \cdots & w_n^0 \\ w_n^0 & w_n^1 & w_n^2 & w_n^3 & \cdots & w_n^{n-1} \\ w_n^0 & w_n^2 & w_n^4 & w_n^6 & \cdots & w_n^{2(n-1)} \\ w_n^0 & w_n^3 & w_n^6 & w_n^9 & \cdots & w_n^{3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ w_n^0 & w_n^{n-1} & w_n^{2(n-1)} & w_n^{3(n-1)} & \cdots & w_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

где  $y_i$  — Фурье-образ многочлена  $A(x)$ .

Фактически нам надо найти обратное преобразование. Магическим образом обратная матрица к квадратной матрице выглядит почти также:

$$\frac{1}{n} \begin{pmatrix} w_n^0 & w_n^0 & w_n^0 & w_n^0 & \cdots & w_n^0 \\ w_n^0 & w_n^{-1} & w_n^{-2} & w_n^{-3} & \cdots & w_n^{-(n-1)} \\ w_n^0 & w_n^{-2} & w_n^{-4} & w_n^{-6} & \cdots & w_n^{-2(n-1)} \\ w_n^0 & w_n^{-3} & w_n^{-6} & w_n^{-9} & \cdots & w_n^{-3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ w_n^0 & w_n^{-(n-1)} & w_n^{-2(n-1)} & w_n^{-3(n-1)} & \cdots & w_n^{-(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

Откуда получаем:  $a_k = \frac{1}{n} \sum_{j=0}^{n-1} y_j w_n^{-kj}$ .

Теперь напомним псевдокод обратного алгоритма:

---

**Algorithm 2** FFT\_inverted
 

---

```

1: function FFT_INVERTED( $A$ )  $\triangleright A$  — Фурье-образ, возвращает коэффициенты многочлена
2:    $n \leftarrow \text{length}(A)$ 
3:   if  $n == 1$  then
4:     return  $A$ 
5:    $A_l \leftarrow \langle a_0, a_2, \dots, a_{n-2} \rangle$ 
6:    $A_r \leftarrow \langle a_1, a_3, \dots, a_{n-1} \rangle$ 
7:    $\hat{A}_l \leftarrow \text{FFT\_inverted}(A_l)$ 
8:    $\hat{A}_r \leftarrow \text{FFT\_inverted}(A_r)$ 
9:   for  $k \leftarrow 0$  to  $\frac{n}{2} - 1$  do
10:     $A[k] \leftarrow \hat{A}_l[k] + e^{i\frac{-2\pi k}{n}} \hat{A}_r[k]$   $\triangleright$  Здесь угол идёт с минусом
11:     $A[k + \frac{n}{2}] \leftarrow \hat{A}_l[k] - e^{i\frac{-2\pi k}{n}} \hat{A}_r[k]$ 
12:     $A[k] \leftarrow A[k]/2$   $\triangleright$  Поделим на  $2 \log n$  раз, а значит поделим на  $n$  в итоге
13:     $A[k + \frac{n}{2}] \leftarrow A[k + \frac{n}{2}]/2$   $\triangleright$  Аналогично строчке выше
14:  return  $A$ 
```

---