Лекция 16

Вычисление дискретного преобразования Фурье и время работы алгоритма Шёнхаге-Штрассена

(Конспект: М. Кузнецов)

16.1 Анализ алгоритма Шёнхаге-Штрассена

Хочется оценить время работы алгоритма.

Лемма 16.1. *ДПФ можно вычислить за время $O(bl\log b)$.*

Мы докажем эту лемму во второй части лекции.

Итак, рекуррентное соотношение на время работы алгоритма Шёнхаге-Штрассена:

$$\begin{aligned} T(n) \leqslant bT(2l) &\text{ (рекурсивные вызовы)}+ \\ &O(bl\log b) \text{ (время на ДПФ)}+ \\ &\text{дешевые операции (сложение, битовые сдвиги)}+ \\ &O((3b\log b)^{\log_2 3}) \text{ (это время работы «простого» алгоритма,} \\ &\qquad\qquad\text{но оно спрячется в } O(bl\log b)). \end{aligned}$$

Итого,

$$T(n) \leqslant bT(2l) + cbl\log b = bT(2l) + cn\log n.$$

Положим $T'(n) = T(n)/n$. Тогда

$$T'(n) \leqslant 2T'(4\sqrt{n}) + c\log n.$$

Докажем, что $T'(n) = O(\log n\log\log n)$ (т.е. $T(n) = O(n\log n\log\log n)$ — почти линейное время).

Доказываем по индукции; пусть для маленьких n выполняется $T'(n) \leqslant c' \log n \log\log n$. Индукционный переход происходит от $4\sqrt{n}$ к n:

$$T'(n) \leqslant 2c'(2+\log n/2)\log(2+\log n/2) + c\log n \leqslant$$
$$c\log n + 4c'\log(2/3) + 4c'\log\log n + c'\log(2/3)\log n + c'\log n\log\log n.$$

За счет увеличения c' по сравнению с c можно добиться, чтобы слагаемое $c'\log(2/3)\log n$ (отрицательное) стало по абсолютной величине больше, чем три первые (положительные) слагаемые; а оставшееся слагаемое — это то, чем мы оцениваем.

16.2 Вычисление ДПФ

Научимся вычислять ДПФ (вычисление обратного преобразования производится аналогично — упражнение). Нам нужно вычислять некоторый многочлен $p(x)$ во многих точках вида w^k, где w — степень двойки (именно, $2^{4l/b}$), а вычисление ведется по модулю $2^{2l}+1$.

Для того, чтобы вычислить значение в точке x_0, достаточно поделить на $x - x_0$ с остатком:

$$p(x) = q(x)(x-x_0) + r(x), \qquad r(x) = \mathrm{const}\ (\text{это и есть } p(x_0)).$$

Пусть $p(x) = a_0 + a_1 x + \ldots + a_{n-1}x^{n-1}$. Тогда

$$(F(a))_i = \sum_{j=0}^{n-1} (w^i)^j a_j = p(w^i).$$

Значит, надо рассмотреть $p \bmod (x - w^i)$.

Заметим, что если $r = p \bmod q$ и $q'|q$, то $p \bmod q' = r \bmod q'$. Поэтому остатки $\bmod\ x - w^i$ можно вычислять так: сначала вычислить остатки $\bmod$ произведений таких $x - w^i$ для разных i, потом остатки этих остатков $\bmod$ меньших произведений, и т. д.:

```
                          n-1    i
                     p mod П( x-w )
                          i=0
                          /\
                         /  \
                         i                     i
             p mod П( x-w )        p mod П( x-w )
(одна половина возможных i)        (другая половина)
                    / \           / \
                   /   \         /   \
                                                      i      j
                                         p mod (( x-w )( x-w ))
                                             (конкретные пары)
            /\ /\  /\ /\  /\  /\  /\  /\             /\
                                                   i              j
                ...                    p mod ( x-w )  p mod ( x-w )
```

Будем разбивать не на $i \leqslant n/2$, $i > n/2$, а аккуратно: так, чтобы произведения получались красивыми.

Лемма 16.2 (к лемме 16.1). *Можно сгруппировать так, чтобы все произведения* $\prod(x - w^i)$ *в дереве имели вид* $x^t - w^s$.

Задача 16.1. Доказать лемму 16.1.

Указание. Пусть $n = 2^k$. Определим $\mathrm{rev}(j) = d_{k-1} \dots d_0$, где $d_0 \dots d_{k-1}$ — битовое представление числа j. Тогда можно доказать, что

$$\prod_{j=l}^{l+2^m-1} (x - w^{\mathrm{rev}(j)}) = x^{2^m} - w^{\mathrm{rev}(l/2^m)}.$$

(Здесь m — номер уровня в дереве.) □

Итак, какие же вычисления надо производить?

$$\sum_{i=0}^{2t-1} a_i x^i \bmod (x^t - w^s) = \sum_{j=0}^{t-1} (a_j + w^s a_{j+t}) x^j$$

(первая часть, $0 \leqslant j \leqslant t-1$, — без изменений, а во второй вместо x^t заменили на w^s). Все, что здесь надо делать — сложение и умножение на степень двойки.

Теперь можно доказать и лемму 16.1, подсчитав количество операций, выполняемых в этом дереве: на каждом уровне дерева — $O(b)$ операций, каждая из которых занимает время $O(l)$; количество уровней — $\log b$.