1. *Полиномы и их представления*

Функция вида $P(x) = a_0 + a_1x + \dots a_{n-1}x^{n-1}$ называется многочленом или полиномом степени $n-1$. Здесь коэффициенты a_i и аргумент x вообще говоря могут быть комплексными. Многочлену можно сопоставить массив его коэффициентов $[a_0, a_1, \dots, a_{n-1}]$. Будем говорить, что это представление многочлена его коэффициентами. Легко заметить, что если добивать этот массив нулями справа, то он будет задавать тот же многочлен, что и раньше.

Дальше будем считать, что в нашей модели вычислений комплексные числа хранятся с абсолютной точностью и арифметические операции с ними выполняются за $O(1)$.

Сколько в таком представлении стоит операция сложения двух многочленов? Нам приходят на вход два массива $[a_0, \dots, a_{l-1}]$ и $[b_0, \dots, b_{m-1}]$. Если добить массив меньшей длины нулями до длины второго и сложить их покомпонентно, то мы получим массив коэффициентов, который задает полином, являющийся суммой полиномов, задаваемых исходными массивами. Таким образом мы потратим $O(l+m)$ арифметических операций.

А сколько стоит перемножить два многочлена, заданных коэффициентами? Если добить массивы нулями до длины $l+m-1$, то

$$(a_0 + a_1x + \dots + a_{l-1}x^{l-1})(b_0 + b_1x + \dots + b_{m-1}x^{m-1}) = \sum_{k=0}^{l+m-2} \left(\sum_{i=0}^{k} a_i b_{k-i} \right) x^k.$$

Тривиальный алгоритм, который вычисляет все коэффициенты многочлена произведния, как свертки $\sum_{i=0}^{k} a_i b_{k-i}$, выполняет $O((l+m)^2)$ арифметических операций.

Многочлен $(n-1)$-ой степени можно однозначно задавать не только его коэффициентами, но и его значениями в n (или больше) различных точках. Действительно, два разных многочлена не могут иметь одинаковые значения в n различных точках, иначе их разность степени не больше $n-1$ будет иметь n корней, что противоречит известной теореме, о том, что многочлен степени d не может иметь больше d корней. Будем говорить, что это представление многочлена его значениями.

Если зафиксировать какие-нибудь $2n-1$ точек $z_0 \dots, z_{2n-2}$, то сколько будет стоить операция перемножения двух многочленов $P(x)$ и $Q(x)$ степени $n-1$, в представлении значениями в заданных точках?

$$[P(z_0), P(z_1), \dots, P(z_{2n-2})] \cdot [Q(z_0), \dots Q(z_{2n-2})] = [(P \cdot Q)(z_0), \dots, (P \cdot Q)(z_{2n-2})],$$

то есть значения произведения полиномов равны произведениям значений. Степень многочлена $P{\cdot}Q$ не превышает $2n-2$ поэтому он будет задаваться однозначно. Получается, что в этом представлении достаточно $O(n)$ операций.

Идея алгоритма эффективного перемножения многочленов в представлении **коэффициентами** состоит в следующем. Оказывается, что при удачном выборе точек переход из представления коэффициентами в представление значениями и обратно можно осуществлять за $O(n \log n)$ арифметических операций. Тогда можно перейти от представления коэффициентами в представление значениями, в нем перемножить многочлены за линию, а затем перейти обратно.

2. *Комплексные корни из единицы*

   Посмотрите картинки и свойства на вики.

   Комплексные корни степени n из единицы мы будем обозначать $\omega_n^0 = 1, \omega_n^1 = \omega_n, \dots, \omega_n^{n-1}$, где $\omega_n = e^{i2\pi/n}$.

3. *Дискретное преобразование Фурье*

   Дискретное преобразование Фурье — это преобразование пространства $\mathbb{C}^n$, которое отображает набор из n чисел в значения многочлена с этими коэффициентами в комплексных корнях из единицы n-ой степени:

$$\mathbf{DFT} : (a_0, a_1, \dots, a_{n-1}) \mapsto (y_0, y_1, \dots, y_{n-1}) = (P(1), P(\omega_n), \dots, P(\omega_n^{n-1})),$$

   где $P(x) = a_0 + a_1 x + \dots a_{n-1} x^{n-1}$. Обозначается $\mathbf{DFT}[a_0, a_1, \dots, a_{n-1}]$.

4. *Быстрое преобразование Фурье*

   Оказывается **DFT** можно вычислять, производя $O(n \log n)$ арифметических операций. Алгоритм называется быстрое преобразование Фурье или **FFT** (Fast Fourier Transform). Идея – разделяй и властвуй. Для простоты считаем, что n – степень двойки (для алгоритма перемножения многочленов нам будет этого достаточно).

$$\begin{aligned}
y_k = P(\omega_n^k) = a_0 + a_1\omega_n^k + a_2(\omega_n^k)^2 + \dots + a_{n-1}(\omega_n^k)^{n-1} = \\
= (a_0 + a_2(\omega_n^k)^2 + \dots + a_{n-2}(\omega_n^k)^{n-2}) + \omega_n^k(a_1 + a_3(\omega_n^k)^2 + \dots + a_{n-1}(\omega_n^k)^{n-2}) = \\
= (a_0 + a_2(\omega_n^{2k}) + \dots + a_{n-2}(\omega_n^{2k})^{(\frac{n}{2}-1)}) + \omega_n^k(a_1 + a_3(\omega_n^{2k}) + \dots + a_{n-1}(\omega_n^{2k})^{(\frac{n}{2}-1)}) = \\
= P_1(\omega_n^{2k}) + \omega_n^k P_2(\omega_n^{2k}),
\end{aligned}$$

   где

$$P_1(x) = a_0 + a_2 x + \dots + a_{n-2} x^{(\frac{n}{2}-1)},$$
$$P_2(x) = a_1 + a_3 x + \dots a_{n-1} x^{(\frac{n}{2}-1)}.$$

   В то время как k пробегает значения $0, 1, \dots, n-1$, ω_n^{2k} **дважды** пробегает комплексные корни из единицы степени $n/2$, то есть числа $1, \omega_{n/2}, \dots, \omega_{n/2}^{(\frac{n}{2}-1)}$ (помедитируйте над картинкой). Таким образом, мы можем вычислить все

$$P(\omega_n^k) = \begin{cases} P_1(\omega_{n/2}^k) + \omega_n^k P_2(\omega_{n/2}^k), & k = 0, 1, \dots, (\frac{n}{2}-1) \\ P_1(\omega_{n/2}^{(k-\frac{n}{2})}) + \omega_n^k P_2(\omega_{n/2}^{(k-\frac{n}{2})})), & k = \frac{n}{2}, \dots, (n-1) \end{cases} \quad (1)$$

если знаем все числа

$$P_1(\omega_{n/2}^l) \text{ и } P_2(\omega_{n/2}^l), \quad l = 1, 2, \ldots, \left(\tfrac{n}{2} - 1\right).$$

А эти числа являются $\mathbf{DFT}[a_0, a_2, \ldots, a_{n-2}]$ и $\mathbf{DFT}[a_1, a_3, \ldots, a_{n-1}]$!

Таким образом алгоритм вычисления $\mathbf{DFT}[a_0, a_1, \ldots, a_{n-1}]$ заключается в вычислении $\mathbf{DFT}[a_0, a_2, \ldots, a_{n-2}]$ и $\mathbf{DFT}[a_1, a_3, \ldots, a_{n-1}]$, а затем применении формул (1). Сложность записывается рекуррентой

$$T(n) = 2T(n/2) + O(n) \quad \Longrightarrow \quad T(n) = O(n \log n).$$

5. *Обратное дискретное преобразование Фурье*

Мы можем эффективно переходить от коэффициентов многочлена к значениям в комплексных корнях из единицы. Оказывается можно и обратно.

Для этого нужно заметить, что **DFT** — линейное преобразование $\mathbb{C}^n$. Действительно, его можно записать в виде

$$Fa = \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \ldots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)^2} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix} = y.$$

Утверждение.

$$F^{-1} = \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n^{-1} & \omega_n^{-2} & \ldots & \omega_n^{-(n-1)} \\ 1 & \omega_n^{-2} & \omega_n^{-4} & \ldots & \omega_n^{-2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \ldots & \omega_n^{-(n-1)^2} \end{pmatrix}$$

Проверьте это самостоятельно, умножив i-ый столбец одной матрицы на j-ую строчку другой.

Таким образом **DFT** обратимо и более того обратное преобразование получается заменой ω_n на ω_n^{-1}. То есть алгоритм вычисления обратного преобразования получается из алгоритма **FFT** заменой во всех вычислениях ω_n на ω_n^{-1} и имеет такую же сложность.

6. *Быстрое перемножение многочленов*

Теперь можно предложить алгоритм перемножения многочленов заданных коэффициентами $[a_0, \ldots, a_{l-1}]$ и $[b_0, \ldots, b_{m-1}]$.

1. Добиваем нулями массивы $[a_0, \ldots, a_{l-1}]$, $[b_0, \ldots, b_{m-1}]$ до длины n равной ближайшей сверху степени двойки к числу $l+m-1$. Полученные массивы обозначим $[a_0, \ldots, a_{n-1}]$ и $[b_0, \ldots, b_{n-1}]$.
2. Вычисляем $\mathbf{DFT}^{-1}[\mathbf{DFT}[a_0, \ldots, a_{n-1}] \cdot \mathbf{DFT}[b_0, \ldots, b_{n-1}]]$.