多项式与生成函数

翁文涛

中山纪念中学

Contents

前言

大家好，今天我们讲一些有关多项式和生成函数的基础知识。如果你已经有一定的知识水平请不要干扰别人听课，就当做是复习一遍就好。不懂的同学也不要中途弃疗，因为今天讲的是基础知识，还是可以听懂的。

如果在讲课途中有什么不太明白的同学可以举手提问。讲课中会有一些例题，欢迎大家交流做法。

一般生成函数（OGF）

对于一个序列 a，我们定义

$$A(x)=\sum_{i\geq 0}a_i x^i$$

为 a 的生成函数。

比如说当序列 a 为 $a_i=1$ 时，$A(x)=1+x^1+x^2+x^3+\cdots=\frac{1}{1-x}$。我们在考虑母函数的问题时是不需要将 x 代入的，因此不需要考虑函数的收敛。

一种更好的理解是，设当前有一个物品集合 A，其中大小为 i 的物品有 A_i 个，那么对于 A 的生成函数 $A(x)=\sum_{i\geq 0}A_i x^i$。接下来基本上都会用这种模型。

操作

假设现在有两个物品集合 A, B，那么有：

- 加法

  对于一个集合 $A + B$，有 $(A + B)(x) = \sum_{i \geq 0} (A_i + B_i) x^i$

- 乘法

  对于 A, B 的笛卡尔积 $A \times B$，有

  $(A \times B)(x) = \sum_{i \geq 0} \left(\sum_{j=0}^{i} A_j B_{i-j} \right) x^i$。

  两个集合 A, B 的笛卡尔积 C，C 中的元素为二元组 (a_i, b_i)，其中 $a_i \in A, b_i \in B$，并且 (a_i, b_i) 的大小定义为 a_i 的大小加上 b_i 的大小。

生成函数的组合意义

我们之前已经知道了生成函数 $\frac{1}{1-x}=1+x+x^2+\cdots$，那么接下来我们可以尝试求一下 $\frac{1}{(1-x)^m}$ 对应的序列是什么。

$\frac{1}{(1-x)^m}=\left(\frac{1}{1-x}\right)^m$，因为 $\frac{1}{1-x}=1+x+x^2+\cdots$，可以想象成是有一个物品集合 A，其中每种权值都恰好有 1 个。那么 $[x^n]A^m(x)$ 相当于从 m 个 A 集合中各抽出来一个物品，并且物品的权值和恰好为 n 的方案数，最后其实就相当于将 n 拆分成 m 个非负整数的方案数，就恰好为 $\binom{n+m-1}{n}$。

因此，

$$\frac{1}{(1-x)^m}=\sum_{i\geq 0}\binom{i+m-1}{m-1}x^i$$

基础习题

求出序列 $A = \{0, 1, 4, 9, \cdots, n^2, \cdots\}$ 的生成函数。

令 $F(x)$ 为序列 A 的生成函数，那么有：

$$
\begin{aligned}
F(x) &= \sum_{i\geq 0} i^2 x^i \\
xF(x) &= \sum_{i>0} (i-1)^2 x^i \\
F(x)(1-x) &= \sum_{i>0} (2i-1) x^i \\
&= 2\sum_{i\geq 0} i x^i - \frac{x}{1-x} \\
&= \frac{2x}{(1-x)^2} - \frac{x}{1-x} \\
&= \frac{x(x+1)}{(1-x)^2} \\
F(x) &= \frac{x(x+1)}{(1-x)^3}
\end{aligned}
$$

经典例题

- 现在有 A, B, C, D 四种水果，每种都有无限个。现在要求拿出恰好 N 个水果，但要求水果 A 要拿出恰好偶数个，B 的个数是 5 的倍数，C 最多只能拿 4 个，D 最多拿 1 个，问最终的方案数是多少。

经典例题

- 现在有 A, B, C, D 四种水果，每种都有无限个。现在要求拿出恰好 N 个水果，但要求水果 A 要拿出恰好偶数个，B 的个数是 5 的倍数，C 最多只能拿 4 个，D 最多拿 1 个，问最终的方案数是多少。
- 请尽量用生成函数来解决。~~假如你想找规律，我也没办法。~~

经典例题

- 现在有 A, B, C, D 四种水果，每种都有无限个。现在要求拿出恰好 N 个水果，但要求水果 A 要拿出恰好偶数个，B 的个数是 5 的倍数，C 最多只能拿 4 个，D 最多拿 1 个，问最终的方案数是多少。
- 请尽量用生成函数来解决。~~假如你想找规律，我也没办法。~~
- 构造出每种水果对应的生成函数乘一乘？

经典例题

令 $g(x)$ 为最终的生成函数，那么有：

$$\begin{aligned} g(x) &= (1+x^2+x^4+x^6+\cdots)\times(1+x^5+x^{10}+\cdots) \\ &\quad \times(1+x+x^2+x^3+x^4)\times(1+x) \\ &= \frac{1}{1-x^2}\times\frac{1}{1-x^5}\times(1+x+x^2+x^3+x^4)\times(1+x) \\ &= \frac{1}{1-x}\times\frac{1}{1-x} \\ &= \frac{1}{(1-x)^2} \end{aligned}$$

套用之前的公式，有 $g(x)=\sum_{i\geq 0}(i+1)x^i$，那么 $[x^n]g(x)=n+1$ 了。

组合数题

试证明：

$$\sum_{j=1}^{n}\sum_{k=j+1}^{n}\binom{n}{j}\binom{n}{k}(k-j)$$

$$=\frac{n\binom{2n}{n}}{2}$$

$$\sum_{j=1}^{n}\sum_{k=j+1}^{n}\binom{n}{j}\binom{n}{k}(k-j)$$

$$=\sum_{r=1}^{n} r\sum_{k-j=r}\binom{n}{j}\binom{n}{k} \to [x^{r}]((1+\frac{1}{x})^{n}(1+x)^{n})$$

$$=\sum_{r=1}^{n} r\binom{2n}{n+r}$$

$$=\sum_{r=1}^{n} r\binom{2n}{n-r}$$

$$=n\sum_{r=1}^{n}\binom{2n}{n-r}-\sum_{r=1}^{n}(n-r)\binom{2n}{n-r}$$

$$=\frac{n\times\left[2^{2n}-\binom{2n}{n}\right]}{2}-\frac{2n\times\left[2^{2n-1}-2\binom{2n-1}{n-1}\right]}{2}=\frac{n\binom{2n}{n}}{2}$$

带标号对象的拼接

OGF 在解决组合问题时是非常方便的。但在 OGF 中，我们存的是某种物品的个数，也就是说同种物品是无标号的，那么当物品有标号时就不太方便了。比如说现在有两个排列 A, B，我们需要将他们拼在一起，那么我们是不能改变他们原本内部的顺序，因此，最终的方案数就是 $\binom{|A|+|B|}{|A|}$。

指数型生成函数（EGF）

对于一个序列 A_i，其指数生成函数为

$$A(x)=\sum_{i\geq 0}\frac{A_i}{i!}x^i$$

也就是说，在实际存储时，我们不存储 A_i，而是存储 $\frac{A_i}{i!}$，最终输出时才将 $i!$ 乘上去。

操作

考虑现在将两个有标号的对象拼接在一起，设分别为 A, B，令 $C = A \cdot B$，那么有：

$$
\begin{aligned}
C_i &= \sum_{j=0}^{i} \binom{i}{j} A_j B_{i-j} \\
&= i! \left(\sum_{j=0}^{i} [x^j] A(x) [x^{i-j}] B(x) \right) \\
C(x) &= \sum_{i \geq 0} C_i \frac{x^i}{i!} \\
&= \sum_{i \geq 0} \left(\sum_{j=0}^{i} [x^j] A(x) [x^{i-j}] B(x) \right) x^i
\end{aligned}
$$

因此，C 的 EGF 恰好就是 A, B 的 EGF 的卷积。

常用 EGF

- $e^x = \sum_{i \geq 0} \frac{x^i}{i!}$

常用 EGF

- $e^x = \sum_{i \geq 0} \frac{x^i}{i!}$
- $\frac{e^x + e^{-x}}{2} = \sum_{i \bmod 2 = 0} \frac{x^i}{i}$

常用 EGF

- $e^x = \sum_{i \geq 0} \frac{x^i}{i!}$
- $\frac{e^x + e^{-x}}{2} = \sum_{i \bmod 2 = 0} \frac{x^i}{i}$
- $\frac{e^x - e^{-x}}{2} = \sum_{i \bmod 2 = 1} \frac{x^i}{i}$

集合的 EGF

- 考虑现在一个集合 S 是由组合对象 A 组成的，那么由于集合的元素间是无序的，因此 $S = \sum_{i \geq 0} \frac{A^i}{i!} = e^A$

集合的 EGF

- 考虑现在一个集合 S 是由组合对象 A 组成的，那么由于集合的元素间是无序的，因此 $S = \sum_{i\geq 0} \frac{A^i}{i!} = e^A$
- 举个例子，令 $f(x)$ 为轮换的生成函数，$g(x)$ 为置换的生成函数。那么有 $[x^n]f(x) = \frac{(n-1)!}{n!} = \frac{1}{n}, (n > 0)$，因为 $g(x) = e^{f(x)} = e^{-ln(1-x)} = \frac{1}{1-x}$，因此 $[x^n]g(x) = 1$，所以长度为 n 的置换数恰好就为 $n!$。

EGF 的应用

- 给定集合 S，求有多少长度为 n 的置换，使得其所有轮换的长度都属于集合 S。

EGF 的应用

- 给定集合 S，求有多少长度为 n 的置换，使得其所有轮换的长度都属于集合 S。
- 令 $f(x) = \sum_{i \in S} \frac{1}{i} x^i$，那么求出 $[x^n] exp(f(x))$ 即可。

EGF 的应用

- 给定集合 S，求有多少长度为 n 的置换，使得其所有轮换的长度都属于集合 S。
- 令 $f(x) = \sum_{i \in S} \frac{1}{i} x^i$，那么求出 $[x^n] exp(f(x))$ 即可。
- 求出恰好包含 n 个点的有标号无向联通图数量。（要求无重边自环）

EGF 的应用

- 给定集合 S，求有多少长度为 n 的置换，使得其所有轮换的长度都属于集合 S。
- 令 $f(x)=\sum_{i\in S}\frac{1}{i}x^i$，那么求出 $[x^n]exp(f(x))$ 即可。
- 求出恰好包含 n 个点的有标号无向联通图数量。（要求无重边自环）
- 令 $G(x)$ 表示无向图数的 EGF，$C(x)$ 表示无向联通图的 EGF，那么有

$$G(x)=\sum_{i\geq 0}2^{\binom{i}{2}}$$

$$G(x)=e^{C(x)}$$

$$C(x)=ln(G(x))$$

计算伯努利数[1]

- 给定 n，求出 $B_0, \cdots, B_n$ 对 998244353 取模的值，其中 B_i 满足：

$$B_0 = 0$$

$$\sum_{i=0}^{n} \binom{n+1}{i} B_i = 0$$

[1]在算 k 次幂和时要用到

计算伯努利数[1]

- 给定 n，求出 $B_0, \cdots, B_n$ 对 998244353 取模的值，其中 B_i 满足：

$$B_0 = 0$$

$$\sum_{i=0}^{n} \binom{n+1}{i} B_i = 0$$

-

$$B \times \frac{e^x - 1}{x} = B_0 = 1$$

$$B = \frac{x}{e^x - 1}$$

[1]在算 k 次幂和时要用到

51nod 1728 不动点

- 现在有 n 个点，每个点有个值 $f(i)$，表示 i 这个点走一步会到 $f(i)$ 这个点。一开始所有的 f 都没有确定。现在给定 k，要求对于每个 i，都满足 i 这个点，走不超过 k 步，能到一个点 c，满足 $f(c) = c$。$n * k <= 2000000, k <= 3$

51nod 1728 不动点

- 现在有 n 个点，每个点有个值 $f(i)$，表示 i 这个点走一步会到 $f(i)$ 这个点。一开始所有的 f 都没有确定。现在给定 k，要求对于每个 i，都满足 i 这个点，走不超过 k 步，能到一个点 c，满足 $f(c) = c$。$n * k <= 2000000, k <= 3$
- 把 $f(i)$ 想象成是 i 指向祖先的边，那么就相当于要统计有多少种森林，满足每个点的深度都不超过 k。

51nod 1728 不动点

- 现在有 n 个点，每个点有个值 $f(i)$，表示 i 这个点走一步会到 $f(i)$ 这个点。一开始所有的 f 都没有确定。现在给定 k，要求对于每个 i，都满足 i 这个点，走不超过 k 步，能到一个点 c，满足 $f(c) = c$。$n * k <= 2000000, k <= 3$
- 把 $f(i)$ 想象成是 i 指向祖先的边，那么就相当于要统计有多少种森林，满足每个点的深度都不超过 k。
- 令 $[x^n]g_k(x)$ 代表深度限制为 k 时，森林点数为 n 的方案数。当 $k = 0$ 时，每个点都必须作为根，则有 $g_0(x) = e^x$，$k = 1$ 时，我们考虑，一棵深度不超过 k 的树，相当于一个深度不超过 $k-1$ 的森林和一个根拼在一起，所以一棵深度不超过 k 的树的生成函数就是 $xg_{k-1}(x)$。然后这个新森林又是若干棵树的集合，所以 $g_k(x) = e^{xg_{k-1}(x)}$。那么我们只需要不停地套用多项式求 exp 的做法

CF623E

- 对于一个长度为 n 的序列 a，令 $b_i = a_1 \text{or} a_2 \cdots \text{or} a_i$，$a$ 被称为合法的当且仅当 b 严格递增。

  给定 n, k，a_i 的值域为 $[0, 2^k)$，问最终有多少个长度为 n 的合法序列 a。答案对 $10^9 + 7$ 取模。

  $n \leq 10^{18}, k \leq 30000$

CF623E

- 对于一个长度为 n 的序列 a，令 $b_i = a_1 \text{or} a_2 \cdots \text{or} a_i$，$a$ 被称为合法的当且仅当 b 严格递增。

  给定 n, k，a_i 的值域为 $[0, 2^k)$，问最终有多少个长度为 n 的合法序列 a。答案对 $10^9 + 7$ 取模。

  $n \leq 10^{18}, k \leq 30000$
- 设 Dp 状态 $f[i][j]$ 表示当前考虑到第 i 个数，二进制位用了 j 个 1 的方案数。那么显然有

$$f[i][j] = \sum_{k=0}^{j-1} \binom{j}{k} f[i-1][k] \times 2^k$$

$$\frac{f[i][j]}{j!} = \sum_{k=0}^{j-1} \frac{f[i-1][k] \times 2^k}{k!} \times \frac{1}{(j-k)!}$$

CF623E

令 $f_i(x)$ 表示 $f[i]$ 的生成函数。那么有 $f_i(x) = f_{i-1}(2x) \times (e^x - 1)$。把最终的式子写出来相当于要求

$$(e^x - 1)(e^{2x} - 1) \cdots (e^{2^n x} - 1) \bmod x^{k+1}$$

计算这条式子可以倍增地求。假设当前已经求出 $(e^{2^0 x} - 1)(e^{2x} - 1) \cdots (e^{2^n x} - 1) \bmod x^{k+1}$，然后我们需要求 $(e^{2^0 x} - 1)(e^{2x} - 1) \cdots (e^{2^{2n+1} x} - 1) \bmod x^{k+1}$，因为我们知道前式的多项式，那么我们直接将 $2^{n+1}x$ 代入多项式中，我们就可以得到后半部分的多项式，再和前面的乘起来就好了。

总的复杂度为 $O(k \log k \log n)$。

常系数齐次线性递推

- 斐波拉契数列

$$f_i = \begin{cases} 0, i = 0 \\ 1, i = 1 \\ f_{i-1} + f_{i-2}, i > 1 \end{cases}$$

常系数齐次线性递推

- 斐波拉契数列

$$f_i = \begin{cases} 0, i = 0 \\ 1, i = 1 \\ f_{i-1} + f_{i-2}, i > 1 \end{cases}$$

- 除了背特征根公式外，怎么手推这种递推方程的通项公式？

常系数齐次线性递推

- 斐波拉契数列

$$f_i = \begin{cases} 0, i = 0 \\ 1, i = 1 \\ f_{i-1} + f_{i-2}, i > 1 \end{cases}$$

- 除了背特征根公式外，怎么手推这种递推方程的通项公式？
- 不妨尝试用生成函数

常系数齐次线性递推 I

- 令 $F(x) = f_0 + f_1 x + f_2 x^2 + \cdots$，那么有

$$
\begin{aligned}
F(x) &= f_0 + f_1 x + f_2 x^2 + \cdots + f_i x^i \cdots \\
xF(x) &= \quad f_0 x + f_1 x^2 + \cdots + f_{i-1} x^i \cdots \\
x^2 F(x) &= \quad\quad f_0 x^2 + \cdots + f_{i-2} x^i \cdots \\
F(x) - xF(x) - x^2 F(x) &= f_0 + (f_1 - f_0)x \\
&= x \\
F(x) &= \frac{x}{1 - x - x^2}
\end{aligned}
$$

常系数齐次线性递推 II

- 我们知道 $\frac{1}{1-ax} = 1 + ax + a^2x^2 + \cdots$，因为 $F(x)$ 是一个有理分式，并且其分母是二次多项式，所以可以知道 $F(x)$ 可以表示成两个形如 $\frac{A}{1-ax}$ 之和，其中 A, a 都是常数。
- $$1 - x - x^2 = -(x + \frac{1+\sqrt{5}}{2})(x + \frac{1-\sqrt{5}}{2})$$
  经过一些运算后可以解得：$F(x) = \frac{1}{\sqrt{5}} * (-\frac{1}{1-\frac{1-\sqrt{5}}{2}x} + \frac{1}{1-\frac{1+\sqrt{5}}{2}x})$，根据之前得到了的结论，我们就可以马上推得：
  $[x^n]F(x) = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^n - (\frac{1-\sqrt{5}}{2})^n]$
- 一般手解方程都会特别痛苦，所以接下来讲些不用解特征根的方法。

特征多项式 I

不妨假设当前有个数列 f_i，有长度为 k 的向量 a 和 C，满足

$$f_i = \begin{cases} a_i, 0 \le i < k \\ \sum_{j=1}^{k} C_j f_{i-j} \end{cases}$$

通过类似于刚刚推斐波拉契数列的生成函数的方法，令 f 的生成函数为 $F(x)$，另有一多项式 $P(x)$，我们可以推出来

$$\begin{aligned} P(x) =& F(x)(1 - C_1 x - C_2 x^2 - \cdots - C_k x^k) \\ =& a_0 + (a_1 - C_1 f_0)x + \cdots + \\ & (a_{k-1} - C_{k-1} a_0 - C_{k-2} a_1 - \cdots - C_1 a_{k-2}) x^{k-1} \end{aligned}$$

特征多项式 II

不妨记 $M(x)=x^k-C_1x^{k-1}-C_2x^{k-2}-\cdots C_k$，那么就有 $F(x)=\frac{P(x)}{M^r(x)}$，并且有 $Deg(P)<k$，$Deg(M)=k$，事实上，对于任意一对 $P(x),M(x)$，假如满足 $Deg(P)<Deg(M)$，那么我们都可以找到一个递推数列 f，满足 f 的生成函数 $F(x)=\frac{P(x)}{M^r(x)}$，转移系数可以由 $M(x)$ 得到，初值向量 a 可以由 P 的定义再通过 $M(x)$ 得到。一般我们称 $M(x)$ 为 f 的特征多项式，事实上你也可以发现，假如我们把 $M(x)$ 的 k 个根全部解出来了，我们同样可以用解斐波拉契数列通项时的做法，把 $F(x)$ 分解为若干个基本分式之和，但这样就显得比较麻烦了。

与多项式取模之间的关系 I

首先给出一个结论：

令 $Q_n(x) = x^n \bmod M(x)$，其中 $M(x)$ 是 f 的特征多项式，那么有 $f_n = \sum_{i=0}^{k-1} [x^i] Q_n(x) a_i$。那么只要我们会多项式取模的快速做法，要求 f_n，我们只需要得到 a 和 $M(x)$，然后做快速幂求出 $x^n \bmod M(x)$ 就可以了。比如说要求斐波拉契数列的第 5 项的值，我们可以求出 $x^5 \bmod x^2 - x - 1 = 5x + 3$，于是 $f_5 = 5$ 了。

证明 I

考虑这个递推的转移矩阵 A，令 $p(\lambda) = |A - \lambda I|$，展开一下就可以得到 $p(\lambda) = \lambda^k - C_1\lambda^{k-1} - \cdots C_k$。由 Hamilton-Cayley 定理可知，$p(A) = 0$，也就是说 $A^k - C_1A^{k-1} - \cdots - A^0 = 0$，即 A 的任何次幂都可以由此次幂的前 K 项线性组合，最后可以推得 A 的任何次幂都是 $A^0, \cdots, A^{k-1}$ 的线性组合。

这样的话我们不妨设

$$A^n = \sum_{i=0}^{k-1} poly_n[i] A^i$$

。在矩阵快速幂时

$$A^{2n} = A^n \times A^n, poly_{2n} = poly_n^2$$

这样我们会得到一个 $2k-1$ 项的多项式，再根据线性递推的关系就可以进一步缩减为一个 k 项的多项式。

而事实上我们根据递推关系缩减的过程就相当于 $\bmod M(x)$，所以，最后可以得到 $poly_n = x^n \bmod M(x)$。再由

$$\begin{aligned} f_n &= (A^n \times a)[k-1] \\ &= \left(\sum_{i=0}^{k-1} poly_n[i] A^i \times a\right)[k-1] \\ &= \sum_{i=0}^{k-1} poly_n[i] \sum_{j=0}^{k-1} A^i[k][j] a[j] \\ &= \sum_{i=0}^{k-1} poly_n[i] a[k-i-1] \end{aligned}$$

注意，上面的 a 是初值向量的翻转（矩阵乘法是倒过来做的）。

对于 $i < k$，$A^i[k-1]$ 这一行只有 $A^i[k-1][k-i-1]$ 这个位置是等于 1 的，其他都为 0。Q.E.D.

一道简单的例题

- 给定 n, k，$f_1, \cdots, f_k$ 满足 $f_1 = x$，$f_k = \frac{x}{1-f_{k-1}}$，求出 $[x^n]f_k$。
  $n \leq 10^9, k \leq 1000$

一道简单的例题

- 给定 n, k，$f_1, \cdots, f_k$ 满足 $f_1 = x$，$f_k = \frac{x}{1-f_{k-1}}$，求出 $[x^n]f_k$。
  $n \le 10^9, k \le 1000$
- 不妨设 $f_i = \frac{a_i}{b_i}$，其中 a_i, b_i 都是某一多项式，那么有 $f_{i+1} = \frac{x}{1-f_i} = \frac{bx}{b-a}$，于是我们可以递推地把 a_i, b_i 求出来。得到了 $f_k = \frac{a_k}{b_k}$ 后，为了使 $Deg(a_k) < Deg(b_k)$，不妨令 $a_k = b_k * R + q(x)$，那么有 $f_k = R + \frac{q(x)}{b_k}$，并且满足 $Deg(q) < Deg(b_k)$，根据之前的结论，我们可以知道 f_k 是一个以 b_k^r 为特征多项式的常系数齐次线性递推竖列 F 的生成函数，反向根据 $q(x)$ 把 F 的初值向量 a 求出后，再求出 $x^n \bmod b_k$ 即可。时间复杂度是 $O(k^2 \log n)$。

51nod 1538 一道难题

给定数组 a，以及两个数 n, m，保证 a 数组的数都是正整数，求：

$$\left(\sum_{\left(\sum_{i=1}^{n} a_i b_i\right)=m} \frac{\left(\sum_{i=1}^{n} bi\right)!}{\prod_{i=1}^{n}(b_i)!} \right) \bmod 104857601$$

$n \le 10^6, m \le 10^{18}, a_i \le 23333$

转换模型

我们不妨看看 $\frac{(\sum_{i=1}^{n} bi)!}{\prod_{i=1}^{n}(b_i)!}$ 的含义是什么。相当于现在有 n 种球，每种球有 b_i 个，同种球之间是等价的，问能不同的排列的方案数。那么我们再看看 $\sum_{i=1}^{n} a_i b_i = m$，相当于每种球都有个权值，要求最后的总权值等于 m。那问题就变为：现在有 n 种球，第 i 种球有权值 a_i，问有多少种排列方案，使得球的总权值为 m。这就是一个很简单递推，令 $f[s]$ 表示总权值为 s 的方案数，那么有

$$f[s] = \sum_{i=1}^{n} f[s - a_i]$$

注意到 $a_i \leq 23333$，所以这个问题就变成了项数不超过 23333 的线性递推，对于 $m > 23333$ 的部分，直接套用之前的做法，令 $c[i]$ 表示权值为 i 的球的种类数，$C(x) = x^{23333} - c[1]x^{23332} - \cdots$，求出 $x^m \bmod C(x)$ 然后和初值向量点积即可。对于求初值向量，我们可以用分治 fft 来求解。

循环卷积

- 一维循环卷积

  假设我们要求出 $A(x), B(x)$ 关于长度为 n 的循环卷积，那么只需要求出 $DFT_n(A(x))$ 和 $DFT_n(B(x))$ 两者点积一下再 Idft 回去就好了。

循环卷积

- 一维循环卷积

  假设我们要求出 $A(x), B(x)$ 关于长度为 n 的循环卷积，那么只需要求出 $DFT_n(A(x))$ 和 $DFT_n(B(x))$ 两者点积一下再 Idft 回去就好了。

- 二维循环卷积

  现在有两个均为 $r \times r$ 的矩阵 A, B，给定 n, m，定义

$$(A \otimes_{n,m} B)[i][j] = \sum_{(x+y) \bmod n = i} \sum_{(u+v) \bmod n = j} A[x][u]B[y][v]$$

  令

$$DFT_{n,m}(A) = \sum_{i=0}^{r-1} \sum_{j=0}^{r-1} w_n^i w_m^j A[i][j]$$

- 那么可以知道

$$DFT_{n,m}(A \otimes_{n,m} B) = DFT_{n,m}(A) \cdot DFT_{n,m}(B)$$

这里矩阵的点积就是对位相乘。考虑给定一个矩阵 A，怎么求出 $DFT_{n,m}A$ 以及 $IDFT_{n,m}A$。先考虑怎么求 $DFT_{n,m}A$，我们可以先对 A 的每一行求出其关于长度 n 的 DFT，再在新的矩阵中对每一列求出关于长度 m 的 DFT 即可。要求 IDFT，我们直接把这个过程逆回去就可以了。

- 那么可以知道

$$DFT_{n,m}(A \otimes_{n,m} B) = DFT_{n,m}(A) \cdot DFT_{n,m}(B)$$

这里矩阵的点积就是对位相乘。考虑给定一个矩阵 A，怎么求出 $DFT_{n,m}A$ 以及 $IDFT_{n,m}A$。先考虑怎么求 $DFT_{n,m}A$，我们可以先对 A 的每一行求出其关于长度 n 的 DFT，再在新的矩阵中对每一列求出关于长度 m 的 DFT 即可。要求 IDFT，我们直接把这个过程逆回去就可以了。

- 多维循环卷积

仿照二维循环卷积的思路就好了。

- FWT

  设当前有一个运算 $(A \times B)$，其中 A, B 长度均为 2^k，满足 $(A \times B)[n] = \sum_{i=0}^{2^k-1} A[i]B[n \text{ xor } i]$，那么事实上，我们假如把 A, B 抽象为一个有 k 维，每一维长度都是 2 的矩阵（应该是这么叫的吧），那这个问题就相当于多维循环卷积了。那么我们就分别对 A, B 做 k 维的关于长度 2 的 DFT，然后点积，再倒回去 IDFT 就好了。所以事实上这个进制改成什么进制都是可以的。大概的复杂度就是 $O(kx^k \log x)$。

- FWT

  设当前有一个运算 $(A \times B)$，其中 A, B 长度均为 2^k，满足 $(A \times B)[n] = \sum_{i=0}^{2^k-1} A[i]B[n \text{ xor } i]$，那么事实上，我们假如把 A, B 抽象为一个有 k 维，每一维长度都是 2 的矩阵（应该是这么叫的吧），那这个问题就相当于多维循环卷积了。那么我们就分别对 A, B 做 k 维的关于长度 2 的 DFT，然后点积，再倒回去 IDFT 就好了。所以事实上这个进制改成什么进制都是可以的。大概的复杂度就是 $O(kx^k \log x)$。

- 如何求关于任意 n 的 DFT？

Bluestein’s Algorithm（搬运自毛爷论文）

- 现在的问题就是，给定一个长度为 n 的数组 A，我们需要求出 B，满足 $B[i] = \sum_{j=0}^{n-1} (w_n^i)^j A[j]$。当 n 是 2 的次幂时，直接用经典的蝴蝶变换就好了嘛，但现在 n 是任意常数。

Bluestein’s Algorithm（搬运自毛爷论文）

- 现在的问题就是，给定一个长度为 n 的数组 A，我们需要求出 B，满足 $B[i]=\sum_{j=0}^{n-1}(w_n^i)^jA[j]$。当 n 是 2 的次幂时，直接用经典的蝴蝶变换就好了嘛，但现在 n 是任意常数。
- 下面直接令 $w=w_n$。

$$\begin{aligned}B[i]&=\sum_{j=0}^{n-1}(w^i)^jA[j]\\&=\sum_{j=0}^{n-1}w^{\frac{i^2+j^2-(i-j)^2}{2}}A[j]\\&=w^{\frac{i^2}{2}}\sum_{j=0}^{n-1}w^{\frac{-(i-j)^2}{2}}A[j]w^{\frac{j^2}{2}}\end{aligned}$$

已经大概变成了卷积的形式了，但还不是卷积，不妨设 C，满足：

$$C[i] = w^{\frac{i^2}{2}} \sum_{j=0}^{i} w^{\frac{-(i-j-n)^2}{2}} A[j] w^{\frac{j^2}{2}}$$

那么因为对于 $i \geq n$，$A[i] = 0$，因此 $B[i] = C[i+n]$，所以我们直接用一次正常的卷积就能得到 $DFT_n(A)$ 了！而且复杂度还是 $O(n \log n)$。事实上可以发现，在运算中我们根本没有用到 w 是单位复根这个性质，因此 w 取任何数时这个算法依然是有效的。

CODECHEF TUPLES 的子问题[2]

给定 n 个数 $a_1, \cdots, a_n$，令 $p = 599 \times 601$，定义 $Y[m] = \sum_{i=1}^{n} \sum_{j=1, a_i a_j \bmod p = m}^{n} 1$，求 $Y[0], \cdots, Y[p-1]$。

$n \leq 10^6$

[2]原问题在此基础上套了一个反演

转化成卷积形式

因为 $p = 599 \times 601$，根据中国剩余定理定理，假如我们知道了 $x \bmod 599, x \bmod 601$，那么 $x \bmod p$ 也就确定了。所以不妨设 $A[i][j]$ 表示 $x \bmod 599 = i, x \bmod 601 = j$ 的 $Y[x]$ 的值。注意到转成离散对数后乘法就直接变成了加法了，那么我们直接把 $x \bmod 599$ 和 $x \bmod 601$ 转成离散对数（当然，对于模出来是 0 的情况要特殊处理），就变成了二维循环卷积了。这题中由于 p 是 300000 级别，所以在求 DFT 时要用到 Bluestein’s Algorithm。

CODECHEF BIKE

给定一副 n 个点 m 条边的有向图，每条边的边权是一个二元组 (a_i, b_i)，你身上也有一个二元组 (a, b)，一开始两个权值都是 0。每经过一条边 (a_i, b_i)，二元组会变成 $((a + a_i) \bmod n, (b + b_i) \bmod n - 1)$，另外给定一个 k，对于每个点 i，以及二元组 (x, y)，你要求出从 i 点出发，经过恰好 k 条边回到 i，并且二元组变成了 (x, y) 的方案数。答案对 1163962801。

$n \leq 22, m \leq 10000, k \leq 10^9$

- 把一个 $n \times (n-1)$ 的矩阵设为类型 Mat，Mat 的乘法运算定义为二维循环卷积。那么这个图的转移矩阵事实上就是一个 $n \times n$ 的以 Mat 为值的矩阵，并且可以发现，由于 Mat 的运算满足交换律，分配律还有结合律，所以转移矩阵的乘法同样可以当成正常矩阵乘法来做。现在我们要做的就是加快这个矩阵乘法的效率。

- 把一个 $n \times (n-1)$ 的矩阵设为类型 Mat，Mat 的乘法运算定义为二维循环卷积。那么这个图的转移矩阵事实上就是一个 $n \times n$ 的以 Mat 为值的矩阵，并且可以发现，由于 Mat 的运算满足交换律，分配律还有结合律，所以转移矩阵的乘法同样可以当成正常矩阵乘法来做。现在我们要做的就是加快这个矩阵乘法的效率。
- 因为 $DFT(A \otimes B) = DFT(A) \cdot DFT(B)$，那么假如我们把一开始转移矩阵的所有元素都 DFT 一遍，那么矩阵乘法的时候一次 Mat 的乘法效率就是 $o(n^2)$ 而不是 $o(n^4)$ 了，然后依然进行正常的矩阵快速幂，把最后得到的矩阵再 IDFT 回去就好了。这里有个问题，就是模数好像不太友好，不能 DFT。事实上测试一下可以发现模数是所有的 $n \leq 11$ 的倍数，那么我们只要能求出单位根，DFT 自然也能做，直接把 n 个值带进去求出点值就好了。

生成函数型 Polya 定理

- **染色问题**

  现在有一条长度为 n 的项链，有 m 种颜色，现在要给每颗珠子染上 m 种颜色中的一种，但要求最终第 i 种颜色被恰好染了 c_i 颗珠子，问有多少种本质不同的方案数。两个方案是本质不同的当且仅当无法通过一种方案旋转得到另一种方案。

生成函数型 Polya 定理

- **染色问题**

  现在有一条长度为 n 的项链，有 m 种颜色，现在要给每颗珠子染上 m 种颜色中的一种，但要求最终第 i 种颜色被恰好染了 c_i 颗珠子，问有多少种本质不同的方案数。两个方案是本质不同的当且仅当无法通过一种方案旋转得到另一种方案。

- 只需要给出一个算法即可。有思路的同学都可以讲一讲。

生成函数型 Polya 定理

- 假如颜色没有数量限制，那么这就是一道最基础的 Polya 计数题。根据 Polya 定理，有等价类数等于 $\frac{\left(\sum_{i=1}^{n} m^{(i,n)}\right)}{n}$，但这题有数量限制，那就不能这样算了。

生成函数型 Polya 定理

- 假如颜色没有数量限制，那么这就是一道最基础的 Polya 计数题。根据 Polya 定理，有等价类数等于 $\frac{\left(\sum_{i=1}^{n} m^{(i,n)}\right)}{n}$，但这题有数量限制，那就不能这样算了。
- 我们不妨把每种颜色 i 视为一个变量 x_i，那么对于一个表示选择 p 的置换 P，令 $f_p(x_1, x_2, \cdots, x_m)$ 为对于置换 P 的不动点数的生成函数，$F(x_1, x_2, \cdots, x_m)$ 为等价类的生成函数，那么有

$$f_p(x_1, x_2, \cdots, x_m) = \left(\sum_{i=1}^{m} x_i^{(p,n)}\right)^{n/(p,n)}$$

$$F(x_1, x_2, \cdots, x_m) = \frac{\sum_p f_p(x_1, x_2, \cdots, x_m)}{n}$$

那么具体地来看，我们就相当于求一个有 m 维的矩阵，矩阵每一维的长度就是 c_i，最终要求出 $A[c_1][c_2]\cdots[c_m]$ 的值，像正常多项式乘法那样求就好了。

牛顿迭代

- 设当前有一个函数 g，现在需要解出一个多项式 f 的前 n 项，使得 $g(f) = 0$。

牛顿迭代

- 设当前有一个函数 g，现在需要解出一个多项式 f 的前 n 项，使得 $g(f)=0$。
- 假设现在知道了 f 的前 n 项 f_0，想得到 f 的前 $2n$ 项。

  那么根据泰勒展开，有

$$0=g(f)=g(f_0)+g'(f_0)(f-f_0)+\frac{g''(f_0)(f-f_0)^2}{2}+\cdots$$

$$g(f_0)+g'(f_0)(f-f_0)\equiv g(f)\pmod{x^{2n}}$$

$$f\equiv f_0-\frac{g(f_0)}{g'(f_0)}\pmod{x^{2n}}$$

直接这样子还是没什么前途的，因为我们并不知道多项式复合要怎么做（或者说是怎么高效的做），但在某些特例下牛顿迭代还是很有前途的。

多项式逆元

- 给定一个多项式 $A(x)$，求出一个多项式 $B(x)$ 使得 $A(x) * B(x) \equiv 1 \pmod{x^n}$。

多项式逆元

- 给定一个多项式 $A(x)$，求出一个多项式 $B(x)$ 使得 $A(x) * B(x) \equiv 1 \pmod{x^n}$。
- 考虑和牛顿迭代一样的倍增法。

  假设当前求出 $B_0(x)$ 使得 $A(x) * B_0(x) \equiv 1 \pmod{x^n}$，那么有

$$(A(x) * B_0(x) - 1) \equiv 0 \pmod{x^n}$$

$$(A(x) * B_0(x) - 1)^2 \equiv 0 \pmod{x^{2n}}$$

$$A(x) * (2B_0(x) - A(x)(B_0(x))^2) \equiv 1 \pmod{x^{2n}}$$

  这样迭代下去就好了。时间复杂度

  $T(n) = T(n/2) + O(n \log n) = O(n \log n)$。

多项式求对数

- 给定一个多项式 $A(x)$，求出一个多项式 $B(x) \equiv \ln(A(x)) \pmod{x^n}$

多项式求对数

- 给定一个多项式 $A(x)$，求出一个多项式 $B(x) \equiv \ln(A(x)) \pmod{x^n}$
- 考虑对 $\ln(A(x))$ 求导，最后再积分回来即可。那么有 $\ln(A(x)) = A'(x)\frac{1}{A(x)}$，所以

$$B(x) = \int \frac{A'(x)}{A(x)}\,dx$$

多项式的求导和积分都是十分简单的，因此求对数也可以在 $O(n \log n)$ 的时间完成。

多项式求指数

- 给定一个多项式 $A(x)$，求出一个多项式 $B(x) \equiv \exp(A(x)) \pmod{x^n}$

多项式求指数

- 给定一个多项式 $A(x)$，求出一个多项式 $B(x) \equiv \exp(A(x)) \pmod{x^n}$
- 考虑用牛顿迭代。相当于 $g(x) = \ln(x) - A$，最终要使 $g(B) = 0$。那么有

$$B = B_0 - \frac{g(B_0)}{g'(B_0)}$$

$$B = B_0 - B_0(\ln(B_0) - A)$$

我们已经知道怎么求多项式的对数了，那么求指数时顺便求出对数就好了。时间复杂度也是 $O(n \log n)$。

多项式除法

- 给定一个 n 次多项式 $A(x)$，m 次多项式 $B(x)$，保证 $m < n$，求出 A 除以 B 的商以及余数。也就是求出一个 $n-m$ 次多项式 $Q(x)$ 以及一个低于 m 次的多项式 $Q(x)$，使得 $A(x) = Q(x) \times B(x) + R(x)$。

多项式除法

- 给定一个 n 次多项式 $A(x)$，m 次多项式 $B(x)$，保证 $m < n$，求出 A 除以 B 的商以及余数。也就是求出一个 $n-m$ 次多项式 $Q(x)$ 以及一个低于 m 次的多项式 $Q(x)$，使得 $A(x) = Q(x) \times B(x) + R(x)$。
- 令 $A^r(x)$ 表示 A 的翻转，有 $A^r(x) = x^n(A(\frac{1}{x}))$。那么有

$$A^r(x) = x^n A(\frac{1}{x}) = x^{n-m} Q(\frac{1}{x}) x^m B(\frac{1}{x}) + x^n Q(\frac{1}{x})$$

$$A^r(x) \equiv Q^r(x) B^r(x) \pmod{x^{n-m+1}}$$

因此，求出 A 和 B 的翻转后，$Q^r(x) = \frac{A^r(x)}{B^r(x)} \pmod{x^{n-m+1}}$。求出 Q 后，R 也求出来了。时间复杂度为 $O(n \log n)$。

多项式复合

- 给定两个 n 次多项式 $A(x), B(x)$，求出

$$\sum_{i=0}^{n} [x^i] A(x) B(x)^i \pmod{x^{n+1}}$$

多项式复合

- 给定两个 n 次多项式 $A(x), B(x)$，求出

$$\sum_{i=0}^{n} [x^i] A(x) B(x)^i \pmod{x^{n+1}}$$

- 多项式复合有比较优的做法，但过于复杂，这里我讲一个 $O(n^2 + n\sqrt{n}\log n)$ 的做法。

  令 $L = \sqrt{n}$，那么相当于

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} [x^{i\times L+j}] A(x) B(x)^{i\times L+j}$$

$$\sum_{i=0}^{L-1} B(x)^{i\times L} \sum_{j=0}^{L-1} [x^{i\times L+j}] A(x) B(x)^j$$

多项式复合

那么我们可以用 $O(n\sqrt{n}\log n)$ 的时间求出 $B(x)^{i\times L}$ 以及 $B(x)^j$，其中 $i,j<\sqrt{n}$，然后求的时候暴力求出 $\sum_{j=0}^{L-1}[x^{i\times L+j}]A(x)B(x)^j$，后面这一部分的复杂度总的来说是 $O(n^2)$ 的，那么总复杂度就是 $O(n^2+n\sqrt{n}\log n)$。

多点求值

- 给定一个 n 次多项式 $A(x)$，以及 m 个 x_i，需要求出将 x_i 代入 $A(x)$ 中对 998244353 的值。

多点求值

- 给定一个 n 次多项式 $A(x)$，以及 m 个 x_i，需要求出将 x_i 代入 $A(x)$ 中对 998244353 的值。
- 考虑使用分治算法。假设一个过程为 $Solve(l, r, A)$，表示求出 $i \in [l, r], x_i$ 代入 $A(x)$ 中的值。

  令 $L(x) = \sum_{i=l}^{mid}(x - x_i), R(x) = \sum_{i=mid+1}^{r}(x - x_i)$，接下来我们递归求 $Solve(l, mid, A \bmod L), Solve(mid + 1, r, A \bmod R)$。

  因为每次递归后，A 的次数都与区间长度相同，并且多项式取模的复杂度是 $O(n \log n)$ 的，那么总的复杂度 $T(n) = 2\,T(\frac{n}{2}) + O(n \log n) = O(n \log^2 n)$。

多点求值

- 给定一个 n 次多项式 $A(x)$，以及 m 个 x_i，需要求出将 x_i 代入 $A(x)$ 中对 998244353 的值。
- 考虑使用分治算法。假设一个过程为 $Solve(l, r, A)$，表示求出 $i \in [l, r], x_i$ 代入 $A(x)$ 中的值。

  令 $L(x) = \sum_{i=l}^{mid}(x - x_i), R(x) = \sum_{i=mid+1}^{r}(x - x_i)$，接下来我们递归求 $Solve(l, mid, A \bmod L), Solve(mid + 1, r, A \bmod R)$。

  因为每次递归后，A 的次数都与区间长度相同，并且多项式取模的复杂度是 $O(n \log n)$ 的，那么总的复杂度 $T(n) = 2\,T(\frac{n}{2}) + O(n \log n) = O(n \log^2 n)$。
- 正确性？

  令 $A(x) = B(x)L(x) + R(x)$。那么将 $x_i | i \in [l, mid]$ 代入后，$L(x) = 0$，因此 $A(x) = R(x)$。对于 $x \in (mid, r]$ 同理。

多点插值

- 给定 $n+1$ 个值 $x_0, \cdots, x_n$，以及 $y_0, \cdots, y_n$，求出 n 次多项式 $F(x)$，使得 $F(x_i) = y_i$。

多点插值

- 给定 $n+1$ 个值 $x_0, \cdots, x_n$，以及 $y_0, \cdots, y_n$，求出 n 次多项式 $F(x)$，使得 $F(x_i) = y_i$。
- 考虑拉格朗日插值

$$F(x) = \sum_{i=0}^{n} y_i \frac{\prod_{i \neq j}(x - x_j)}{\prod_{i \neq j}(x_i - x_j)}$$

令 $z_i = \frac{y_i}{\prod_{i \neq j}(x_i - x_j)} \prod_{i \neq j}(x - x_j)$，那么 $F(x) = \sum_{i=0}^{n} z_i$

多点插值

- 给定 $n+1$ 个值 $x_0, \cdots, x_n$，以及 $y_0, \cdots, y_n$，求出 n 次多项式 $F(x)$，使得 $F(x_i) = y_i$。
- 考虑拉格朗日插值

$$F(x) = \sum_{i=0}^{n} y_i \frac{\prod_{i \neq j}(x - x_j)}{\prod_{i \neq j}(x_i - x_j)}$$

令 $z_i = \frac{y_i}{\prod_{i \neq j}(x_i - x_j)} \prod_{i \neq j}(x - x_j)$，那么 $F(x) = \sum_{i=0}^{n} z_i$

- 考虑分治，对于一个区间 $[L, R]$，先把左右只考虑其区间内部贡献的 z_i 的和，那么回溯上去时相当于左边乘上右边的 $\prod(x - x_j)$，右边乘上左边的，然后加起来即可。复杂度 $o(n \log^2 n)$。