生成函数选讲

qwaszx

2022 年 6 月 19 日

符号约定

一般来说 x, z, t, u, v 都可能表示自变量，也即 $f(x)$ 里面的 x。
我们混用 f 和 $f(x)$，具体的含义根据上下文确定。
$y'(x) = y'$ 代表函数 $y(x)$ 的导数，$y^{(n)}(x) = y^{(n)}$ 代表 n 阶导数。
如果对多元函数使用了 F' 记号，那么 x 和 z 优先级更高（不会同时使用这三个）
$\dfrac{\partial F}{\partial x} = \dfrac{\partial}{\partial x}F = \partial_x F = \partial_1 F$，这是 x 是第一个参数的情况，后面可能接自变量。如果自变量里面还有复合，不对内部的复合求导。当然多元函数偏导符号本身就比较有迷惑性，如果不清楚请提问。
$\dfrac{\partial^k}{\partial x^k}F$ 为对 x 的 k 阶偏导。你也许还能见到 $\partial^k_{x^k}F$ 这样的记号。
多数情况下 D 代表求导算子，但不总是这样，请结合上下文。
算子 $\vartheta = xD$ 为 $\vartheta(f) = xf'$

$\sum$ 和 $\prod$ 的求和范围根据上下文，有时不会明确指出。例如在幂级数中有 $i \geq 0$，而在含有二项式系数 $\binom{n}{i}$ 的和式中 i 真的没有范围。
$F^{-1}(x)$ 一般代表复合逆，$F(x)^{-1}$ 则为乘法逆。但不保证总是这样，例如当省略自变量时无法区分两者。幸运的是复合逆出现很少，几乎只在拉格朗日反演的等式左边出现。
$F(x)|_{x=a} = F(a)$，有时 F 很巨大里面还有一些求导什么的，这时这个符号更简洁一些
$F_p = Z_p = Z/pZ$ 表示大小为 p 的有限域，即在 $\mathrm{mod} p$ 下运算 (前面那个等号其实是同构)。$K(x)$ 是关于自变量 x 的有理分式域。
总之千言万语汇成一句话：根据上下文确定，不确定请提问

参考资料？

- cmd 的博客，虽然我也没看过，但我选择相信 cmd（大雾）
- EI 的 csdn 博客，EI 的洛谷博客，以及 EI 的论文/讲课课件（你可以在一些群（比如 uoj 群，LA 群，EI 粉丝群）找到）。How Elegia's mind works?

目录

线性递推

只要问题可以写为 $A(x) = \dfrac{P(x)}{Q(x)}$，其中 $\deg P(x) < \deg Q(x)$，那么这就是一个线性递推（为什么?）

下面设 $m = \deg Q(x)$，$[x^0]Q(x) = 1$

旧的算法可以从矩阵角度得出

$a_n = -\sum_{i=1}^{m} a_{n-i}q_i$，可以写为矩阵形式

$$\begin{bmatrix} a_n \\ \vdots \\ a_{n-m+1} \end{bmatrix} = \begin{bmatrix} -q_1 & -q_2 & \cdots & -q_m \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & \\ 0 & \cdots & 1 & 0 \end{bmatrix} \begin{bmatrix} a_{n-1} \\ \vdots \\ a_{n-m} \end{bmatrix}$$

我们计算 $M^n[a_{m-1} \ \cdots \ a_0]^T$ 就能得到 $[a_{n+m-1} \ \cdots \ a_n]^T$

如果你想手算得到通项，可以把矩阵做 Jordan 分解（然而计算量太大，不是个好办法）

考虑 Cayley-Hamilton 定理，设 M 的特征多项式为 $f(x)$，那么 $f(M) = O$。

于是我们计算 $r(x) = x^n \bmod f(x)$，则 $M^n = r(M)$。而我们可以轻松得到 $M^i\mathbf{v}_0 (i < m)$，这就完成了计算。

手工计算出特征多项式为 $x^m - \sum_{i=1}^{m} q_i x^{m-i}$。整个问题的复杂度为 $O(m \log m \log n)$，暴力取模则为 $O(m^2 \log n)$

存在其不可替代性，例如需要在各项前配上某个系数求和时

新算法直接计算 $[x^n]\frac{P(x)}{Q(x)}$，并且更泛用

其核心在于 $Q(x)Q(-x)$ 的奇次项全为 0

考虑 $\frac{P(x)Q(-x)}{Q(x)Q(-x)}$，其分母为 x^2 的多项式，于是提取 x^n 项系数时只会保留分子的奇次项或偶次项中的一个。

不妨假设保留了分子的奇次项，那么现在分式形如 $\frac{xU(x^2)}{V(x^2)}$

据此可以将 n 减小一半。当 n 充分小时可以直接求逆得到答案，或者干脆让 n 递归到 0 也行。

如果使用 FFT，可以进一步卡常。可以参考 我的博客

可以反过来优化计算 $x^n \bmod Q(x)$

例 (完全背包)

$$[x^N]\prod_{i=1}^{n}\frac{1}{1-x^{c_i}}$$

$N \le 10^{100}, n \le 50, c_i \le 500$，任意模

上下乘 $\prod(1+x^{c_i})$ 可以把分母变成 x^2 的多项式，然后递归。
分子次数始终是 $O(nC)$
注意乘单个 $1+x^{c_i}$ 是 $O(nC)$ 的，于是总复杂度 $O(n^2C\log N)$

例 (在正睿见过的题)

求把 N 拆成 b 的幂的方案数，例如 $N=5,b=2$ 时可以拆成

$1+1+1+1+1$，$1+1+1+2$，$1+2+2$，$1+4$

$N,b\le 10^{18}$

$$[x^N]\frac{1}{(1-x)(1-x^b)(1-x^{b^2})\cdots}$$

上下乘 $\frac{1-x^b}{1-x}$，这是个多项式
分母变成 x^b 的多项式，于是只需要提取分子的所有 $kb+r$ 项，$r = N \bmod b$。递归。始终保持形式

$$[x^N]\frac{P(x)}{(1-x)^c(1-x^b)\cdots}$$

递归一次花费 $O(c^2)$，$c = O(\log N)$，总共 $O(\log N)$ 次递归。

例 (简单算术， zx2003)

链接

其实不是线性递推而是整式递推，且做法和整式递推也没啥关系
有结论 $f(x)^p \equiv f(x^p) \pmod p$（为什么？）
那么可以对 m 做 p 进制拆分，每次计算一个
$[x^k]f(x^p)^{m/p}f(x)^{m \bmod p}C(x)$
提取对应系数后递归到 $k/p, m/p$
预处理所有的 $f(x)^r (0 \le r < p)$

例 (奇怪的题，alpha1022)

链接

读题太费劲了，所以直接把 GF 写出来了：

设 $F(x)=\dfrac{x}{(1-x)^3}$，$G(x,w)=\dfrac{xw(1+w)}{(1-xw)^2}$，求

$$[x^n]\frac{2+F(x)}{1-F(x)G(F(x),w)}$$

出题人写了一大堆阴间拉格朗日反演，讨论了 114514 个细节，得到了一个巨大 $O(n\log n)$ 做法
然而我们直接当成关于 x 的线性递推做，每个 x^i 前面有一个系数 $f_i(w)$。暴力做关于 x 的卷积，系数的乘法用 NTT 加速。每折半一次 $\max\deg f_i$ 的最多乘二。
如果要卡常可以等 n 非常小之后暴力求几项逆。
复杂度 $O(n\log n)$，既没有思维难度也没有代码难度

分式分解

求线性递推通项的一个常用手段是分式分解，例如斐波那契数列 $\frac{1}{1-x-x^2}$，我们将其分解为 $\frac{A}{1-r_1x}+\frac{B}{1-r_2x}$，待定系数算出 A,B 就能知道通项。
我们把它一般化，假设已知 $Q(x)=\prod q_i(x)$，诸 $q_i(x)$ 互质，计算分式分解

$$\frac{P(x)}{Q(x)}=\sum\frac{R_i(x)}{q_i(x)}$$

把右边通分得到 $\frac{\sum V_i(x)R_i(x)}{Q(x)}$，其中 $V_i(x)=Q(x)/q_i(x)$，那么应该有

$$P(x)=\sum V_i(x)R_i(x)$$

注意我们在两边模掉 $q_i(x)$ 会得到

$$P(x)\equiv V_i(x)R_i(x)\pmod{q_i(x)}$$

这是因为其他 $V_j(x)$ 都含有 $q_i(x)$ 因子。

我们可以通过分治计算出所有 $P(x) \bmod q_i(x)$ 和 $V_i(x) \bmod q_i(x)$，然后通过多项式欧几里得解出这个同余方程。常见的情况是 $q_i(x) = (1 - r_i x)^{k_i}$，这时可以简化。我们做换元 $u = 1 - r_i x$，那么 $x = (1 - u)/r_i$，这时方程变为

$$\hat{P}(u) \equiv \hat{V}_i(u)\hat{R}_i(u) \pmod{u^{k_i}}$$

注意 $P \to \hat{P}$ 这个可逆变换是保度数的，所以直接在 $\bmod u^{k_i}$ 下求 $\hat{V}_i$ 的逆即可。
变换和逆变换都可以通过卷积在 $O(k_i \log k_i)$ 时间内完成。

ODE

ODE 的意思是常微分方程，即 $F(x,y,y',\cdots,y^{(n)})=0$
OI 中常用的形式是

$$\sum_{i=0}^{m} f_i(x)y^{(i)}(x)=0$$

其中 $f_i(x)$ 为常数次数（有时是常数项数）的有理分式，m 一般也是常数。当然例外也不少。
我们可以利用上式加速计算 $y(x)$：对 $f_i(x)$ 通分并在两边提取 $[x^n]$，可以得到 y 的系数的一个整式递推。
对于函数 y，如果存在上面那样的微分方程，就称它微分有限 (D-Finite)。
这里常用的一个操作是 $\vartheta=xD$，其作用是在 x^i 前面乘 i。

一些例子

m 次多项式 $\sum_{i=0}^{m} a_i x^i$ 微分有限： $y^{(m+1)} = 0$
幂函数 x^α 微分有限： $xy' = \alpha y$
指数函数 e^x 微分有限： $y' = y$
对数函数 $\ln x$ 微分有限： $xy' = 1$
超几何级数

$$F\left(\begin{matrix} a_1 \cdots a_m \\ b_1 \cdots b_n \end{matrix} \middle| z\right) = \sum_{k\geq 0} \frac{a_1^{\overline{k}} \cdots a_m^{\overline{k}} z^k}{b_1^{\overline{k}} \cdots b_n^{\overline{k}} k!}$$

微分有限：

$$D(\vartheta + b_1 - 1) \cdots (\vartheta + b_n - 1)y = (\vartheta + a_1) \cdots (\vartheta + a_m)y$$

ODE 自动机

有以下结论成立:

- 如果 f, g 微分有限，那么 $f+g$ 微分有限
- 如果 f, g 微分有限，那么 fg 微分有限
- 如果 f 微分有限，g 是代数的（即 g 是某个多项式方程的根），那么 $f \circ g$ 微分有限

我们给出一个构造式的证明，你可以基于此来构建自己的 ODE 自动机（or 手算）

f+g

设 $f^{(n)}=\sum_{i=0}^{n-1}a_i(x)f^{(i)}$， $g^{(m)}=\sum_{i=0}^{m-1}b_i(x)g^{(i)}$
考虑第一个式子，在两边求导并用右边替换 $f^{(n)}$，可以得到

$$f^{(n+1)}=\sum_{i=0}^{n-1}\hat{a}_i(x)f^{(i)}$$

类似可以得到 $f^{(n+k)}$ 关于 $f,\cdots,f^{(n-1)}$ 的线性表示。
于是 $(f+g)^{(k)}$ 总可以被 $f,\cdots,f^{(n-1)},g,\cdots,g^{(m-1)}$ 线性表示。
从而 $f+g,\cdots,(f+g)^{(n+m)}$ 必然线性相关，这就得到了一个 ODE.
实现时，可以依次向线性基中插入 $(f+g)^{(i)}$ 来完成。

fg

想法是一样的，有

$$(fg)^{(k)} = \sum_i \binom{k}{i} f^{(i)} g^{(k-i)}$$

这可以被 $f^{(i)}g^{(j)}, 0 \leq i < n, 0 \leq j < m$ 线性表示，从而存在一个阶数不超过 nm 的微分方程

f∘g

我们先来看看比较简单的情况，即 g 是有理分式。那么有

$$(f\circ g)' = (f'\circ g)g'$$
$$(f\circ g)'' = (f'\circ g)g'' + (f''\circ g)(g')^2$$

以此类推，可以知道

$$(f\circ g)^{(k)} = \sum_{i=0}^{k} c_i(x)(f^{(i)}\circ g)$$

而

$$f^{(n)}\circ g = \sum_{i=0}^{n-1} a_i(g)(f^{(i)}\circ g)$$

也就是说 $f^{(i)}\circ g, 0\le i<n$ 为一组基，于是必有阶数不超过 n 的 ODE.

f∘g

现在考虑一般情况。设 g 满足代数方程 $P(x,g)=0$，其中
$P(x,y)=\sum_{i=0}^{m-1} b_i(x)y^i$
作为基础，有 $(f\circ g)'=(f'\circ g)g'$。我们来看看 g' 如何用 g 表示。有

$$(P(x,g(x)))' = \partial_1 P(x,g(x)) + \partial_2 P(x,g(x))g'(x) = 0$$

这样就有 $g'(x) = -\partial_1 P(x,g)/\partial_2 P(x,g)$。
为了写成关于 g 的多项式，考虑 $P(x,g(x))=0$，我们要解一个同余方程

$$\partial_1 P(x,y) + \partial_2 P(x,y)u(x,y) \equiv 0 \pmod{P(x,y)}$$

可以证明这个总有解。为了防止多解，我们再假设 $P(x,y)$ 关于 y 在 $K(x)$ 上不可约。这样我们可以用多项式欧几里得解出 $u(x,y)$，那么 $g'(x)=u(x,g(x))$。

$f \circ g$

类似有理分式的情况，我们还需要考虑

$$f^{(n)} \circ g = \sum_{i=0}^{n-1} a_i(g)(f^{(i)} \circ g)$$

$a_i(g)$ 的计算依然通过多项式欧几里得完成。
那么我们可以把 $f^{(n+k)} \circ g$ 写为 $g^i(f^{(j)} \circ g), 0 \le i < m, 0 \le j < n$ 的线性组合。从而 $f \circ g$ 的 ODE 阶数不超过 nm。

例 (中国象棋，djq)

$$[x^n](1-x)^{-\frac{1}{2}}\exp\left(\frac{x(1+x)}{2-2x}\right)\frac{\left(\frac{2-x}{2-2x}\right)^k}{k!}\cdot {}_0F_1\left(;k+1;x\left(\frac{2-x}{2-2x}\right)^2\right)$$

除了指着像上面这样一个巨大 GF 说“可以 $O(n)$”（也许是 $O(\sqrt{n}\log n)$ 以外，ODE 还是有一些实际的用途的。我们来看一些例子。

例 (短分式幂)

设 F 为一给定 m 次有理分式（分子分母次数都不超过 m），可以在 $O(nm)$ 时间内计算
$F^\alpha, e^F, \ln F$（如果后两者有定义）的前 n 项

例 (通用测评号)

链接

第一步转化是把选择一个没满的放宽为随机选择一个。显然答案不变。
由期望的线性性，只需要计算某一个框满的概率。计算关于操作次数的 EGF，即

$$F(x) = \frac{n-1}{n}(e^x - S_{a-1}(x))(e^x - S_{b-1}(x))^{n-2} S_{b-1}$$

其中 $S_a(x) = \sum_{i=0}^{a} \frac{x^i}{i!}$。答案即

$$\sum_{i\geq 0} \frac{i!}{n^i}([x^i]F(x))$$

我们先看一下 $F(x) = e^{ax} x^b$ 的时候答案是什么，这是

$$a^{-b} \sum_{i\geq b} \frac{i^{\overline{b}} a^i}{n^i} = b! \frac{n^{-b}}{1-(a/n)^{b+1}}$$

现在我们只要考虑怎么把 $F(x)$ 表示成 $e^{ix}x^j$ 的线性组合。展开二项式，我们需要算 $O(n)$ 个形如

$$A_k(x) = S_a(x)S_b(x)^k$$

的东西。对此我们有

$$\begin{aligned}
(S_a(x)S_b(x)^k)' &= S_a'(x)S_b(x)^k + kS_a(x)S_b(x)^{k-1}S_b(x)' \\
&= (S_a(x) - \frac{x^a}{a!})S_b(x)^k + kS_a(x)S_b(x)^{k-1}(S_b(x) - \frac{x^b}{b!}) \\
&= A_k(x) - \frac{x^a}{a!}S_b(x)^k + kA_k(x) - k\frac{x^b}{b!}A_{k-1}(x)
\end{aligned}$$

$S_b(x)^k$ 也可以同样计算。这样我们就得到一个 $O(n^3)$ 的做法。

例 (珍珠，好像是 laofu)

链接

设有 i 个颜色是奇数个，那么应该有 $(n-i)/2 \ge m$，即 $i \le n-2m$。枚举 i 写 GF：

$$\frac{n!}{D^n 2^D}[x^n] \sum_{i=0}^{\min(D,n-2m)} \binom{D}{i} (e^x - e^{-x})^i (e^x + e^{-x})^{D-i}$$

只需要关心后面这个东西怎么算，注意这是关于 e^x 的函数，抽掉 e^x 得到

$$F(x) = x^{-D} \sum_{i=0}^{N} \binom{D}{i} (x^2-1)^i (x^2+1)^{D-i}$$

后面这个东西显然可以 $O(n \log^2 n)$ 算，大力展开还能 $O(n \log n)$。不过我们不满足于此，我们试着找一个 $O(n)$ 的做法。

首先提出 $(x^2+1)^D$ 得到

$$\sum_{i=0}^{N}\binom{D}{i}\left(\frac{x^2-1}{x^2+1}\right)^i$$

这是关于 $H(x)=\dfrac{x^2-1}{x^2+1}$ 的多项式，我们先来看看

$$F(x)=\sum_{i=0}^{N}\binom{D}{i}x^i$$

这个几乎是 $(1+x)^D$，然而在 N 处截断了。但是我们可以稍加修补：

$$(1+x)F'(x)=DF(x)-(N+1)\binom{D}{N+1}x^{N+1}$$

这个为什么对呢？这个整式递推的阶数为 1，即 f_n 只由 f_{n-1} 决定。且除去我们加的一项外，这个方程是齐次的。所以在 0 到 N 次项它是正确的，在 $N+1$ 次项我们刚好让它为 0，在 $N+1$ 后面的项由于齐次全为 0。这样我们就得到了一个微分方程。

这个想法是 《载谭》 的一个基础

然而我们的最终目标是计算 $G(x)=(x^2+1)^D F(H)$。我们先算 $F(H)$，有 $(F(H))'=F'(H)H'$，于是 $F'(H)=H^{-1}(F(H))'$，这就得到

$$(1+H)H^{-1}(F(H))'=DF(H)-(N+1)\binom{D}{N+1}H^{N+1}$$

还差一点。有
$G'(x)=D(x^2+1)^{D-1}(2x)F(H)+(x^2+1)^D(F(H))'$，记 $U(x)=(x^2+1)^D$，则 $U'=2DxU/(1+x^2)$。考虑

$$(1+H)H^{-1}U(F(H))'=DUF(H)-(N+1)\binom{D}{N+1}H^{N+1}U$$

$$(1+H)H^{-1}U'F(H)=\frac{2Dx(1+H)}{1+x^2}H^{-1}UF(H)$$

上下相加就得到

$$(1+H)H^{-1}G'(x)=\left(\frac{2Dx(1+H)H^{-1}}{1+x^2}+D\right)G(x)+R(x)$$

其中

$$R(x) = -(N+1)\binom{D}{N+1}H^{N+1}U$$

把两边的分母乘一乘就得到一个递推式。$R(x)$ 容易计算。

拉格朗日反演

设 F 是 G 的复合逆，即 $F(G)=x$，$g_0=0,g_1=1$，F,G 都是幂级数，那么有

$$[x^n]H(F)=\frac{1}{n}[x^{n-1}]H'(x)\left(\frac{x}{G}\right)^n$$

以及另类形式

$$[x^n]H(F)=[x^n]H(x)G'(x)\left(\frac{x}{G}\right)^{n+1}$$

后者的优势在于不需要做除法，可以处理 0 次项（0 次项的产生是因为 H 可能有 x^{-k} 这样的项，不过一般 H 都是幂级数）

证明

关键的引理是

$$[x^{-1}]G^nG' = [n = -1]$$

引理的证明是容易的。展开 $H(F) \circ G = H(x)$ 得到

$$\sum_i a_i G^i = H(x)$$

为了凑出引理的形式，我们可以在两边求导后乘 G^{-n}，或者直接在两边乘 $G'G^{-n-1}$，这就得到

$$\sum_i a_i i G^{-n+i-1}G' = H'G^{-n}$$

或者

$$\sum_i a_i G^{-n+i-1}G' = HG^{-n-1}G'$$

提取 $[x^{-1}]$ 并在两边乘 x^{n+1} 就得到两种形式

例

$$[x^n]\frac{1}{1-uF} = [x^n]\frac{1}{1-ux}G'\left(\frac{x}{G}\right)^{n+1}$$

求出后面一串就能得到 u 的系数。这求出了 $[x^n]F^0, \cdots, [x^n]F^m$。如果 F 有常数项，可以写为 $F = \hat{F} + c$ 的形式。其他情况也可以稍作调整。

例 (有标号树)

有标号有根树的 EGF $T(x)$ 满足

$$T(x) = xe^{T(x)}$$

由拉格朗日反演得到 $n![x^n]T(x) = n^{n-1}$

例 (广义二项级数)

设 $B_t(z)$ 满足 $B_t(z) = 1 + zB_t(z)^t$，那么有

$$B_t(z)^r = \sum_{n\geq 0} \frac{r}{tn+r}\binom{tn+r}{n} z^n$$

$$\frac{B_t(z)^r}{1-t+tB_t(z)^{-1}} = \sum_{n\geq 0}\binom{tn+r}{n} z^n$$

需要特别注意 $t=2$ 的情况，此时

$$B_2(z) = \frac{1-\sqrt{1-4z}}{2z}$$

是卡特兰数的 OGF

例 (opencup)

有 R 个红球，B 个蓝球和一个绿球，同色球之间无区别. 将其任意排列，令 l_R, l_B, r_R, r_B 分别为绿球左/右边的红/蓝球数量，定义一个方案的权值为 $\max\{x \in Z | l_R \geq l_B x \ \wedge \ r_R \geq r_B x\}$，求所有方案的权值和. $R \leq 10^{18}, B \leq 10^6$.

题解

奇怪的东西

我们对广义二项级数的一个泛化做一个证明。
设 $F(x)$ 为常数项为 1 的幂级数，
$f_n(x)=[z^n]F(z)^x=[z^n]\exp(x\ln F(z))$，这是关于 x 的 n 次多项式。设幂级数 $\mathcal{F}_t(z)$ 满足

$$\mathcal{F}_t(z)=F(z\mathcal{F}_t(z)^t)$$

我们来提取其系数。设 $G(z)=z\mathcal{F}_t(z)^t$，上式等价于 $G(z)=zF(G(z))^t$。那么由拉格朗日反演得

$$\begin{aligned}[z^n]\mathcal{F}_t(z)^x=&[z^n]F(G(z))^x\\ =&\frac{1}{n}[z^{n-1}](F(z)^x)'F(z)^{tn}\\ =&\frac{x}{tn+x}\frac{1}{n}[z^{n-1}](F(z)^{x+tn})'\\ =&\frac{x}{x+tn}f_n(x+tn)\end{aligned}$$

现在我们知道

$$\mathcal{F}_t(z)^x = \sum_{n\geq 0} \frac{x}{x+tn} f_n(x+tn) z^n$$

借助求导得到

$$\mathcal{F}_t(z)^x + \frac{zt}{x}(\mathcal{F}_t(z)^x)' = \sum_{n\geq 0} f_n(x+tn) z^n$$

另一方面有

$$\mathcal{F}_t(z)' = F'(z\mathcal{F}_t(z)^t)\left(\mathcal{F}_t(z)^t + zt\mathcal{F}_t(z)^{t-1}\mathcal{F}_t(z)'\right)$$

可以解出 $\mathcal{F}_t(z)'$，代回去得到左边等于

$$\sum_{n\geq 0} f_n(x+tn) z^n = \frac{\mathcal{F}_t(z)^x}{1 - zt\mathcal{F}_t(z)^{t-1}F'(z\mathcal{F}_t(z)^t)}$$

取 $F(z) = 1+z$ 得到广义二项级数，$F(z) = e^z$ 得到广义指数级数，$F(z) = \dfrac{z}{1-e^{-z}}$ 得到斯特林多项式

例 (1349F2，xtq)

链接

写 GF 的过程比较复杂，我们直接列出来：

$$F(x)=\frac{e^x-1}{x}, ans=[x^n]\frac{1}{(1-F(x))(1-t(xF(x)))}$$

EI 的另解：

$$[z^n]\frac{t(e^{z(1-t)}-1)}{(1-z)(1-te^{z(1-t)})}$$

先来看看第一个做法，我们找 $w(x) = xF(x)$ 的复合逆，这是 $G(x) = \ln(1+x)$，现在问题是把 $F(x)$ 用 G 表示，注意 $w(G(x)) = x$，所以 $F(x) = (F \circ w)(G)$。现在可以使用拉格朗日反演：

$$ans = [x^n] \frac{1}{(1 - F(w(x)))(1 - tx)} G'(x) \left(\frac{x}{G}\right)^{n+1}$$

因为 F 很简单，所以 $F(w(x))$ 容易 $O(n \log n)$ 计算，最后化成形式

$$[x^n] \frac{1}{1 - tx} H(x)$$

直接展开计算即可

现在来看看 EI 的做法，这条路麻烦一些。这里面最复杂的项应该是 $e^{z(1-t)}$，我们尽力化简它。首先从分子上拆掉：

$$\begin{aligned}&[z^n]\frac{t(e^{z(1-t)}-1)}{(1-z)(1-te^{z(1-t)})}\\=&[z^n]\frac{1}{1-z}\left(\frac{1-t}{1-te^{z(1-t)}}-1\right)\end{aligned}$$

后面的 -1 很好处理，$1-t$ 与 z 无关可以提出来，我们要处理的就是

$$(1-t)[z^n]\frac{1}{(1-z)(1-te^{z(1-t)})}$$

接下来换元 $u=z(1-t)$，就变成

$$(1-t)^{n+1}[u^n]\frac{1-t}{(1-u-t)(1-te^u)}$$

我们最终希望得到 $\dfrac{G(u)}{1-tF(u)}$ 这样方便展开的形式，所以做分式分解：

$$(1-t)^{n+2}[u^n]\left(\frac{\frac{1}{1-u}}{(1-e^u+ue^u)(1-\frac{t}{1-u})}+\frac{e^u}{(1-e^u+ue^u)(1-te^u)}\right)$$

前一项是 $H(u)\dfrac{1}{1-\frac{t}{1-u}}$，提取 $[t^i]$ 得到

$$[u^n]H(u)\frac{1}{(1-u)^i}=\sum_{k\geq 0}\binom{k+i-1}{k}h_{n-k}$$

这是个减法卷积。后一项用拉格朗日反演，取 $F(u)=e^u-1$ 的复合逆 $G(u)=\ln(1+u)$，则化为

$$\frac{u+1}{(-u+G(u)(u+1))(1-t(u+1))}\circ G^{-1}$$

从这里我们可以看到，二元 GF 的诡异之处在于外表上完全不同的形式得到的答案可能是一样的，然而其计算难度完全不同。这与一元 GF 有本质的不同。

例 (斯特林数)

行后缀 S1：$\begin{bmatrix} n \\ n-k \end{bmatrix}$

行后缀 S2：$\begin{Bmatrix} n \\ n-k \end{Bmatrix}$

n 很大，求 $0 \le k \le m$ 的结果

S2 固定下指标的 EGF 为

$$\frac{1}{m!}\left((e^x-1)\right)^m$$

我们要算的是

$$\frac{n!}{(n-k)!}[x^n u^{n-k}]\frac{1}{1-u(e^x-1)}$$

由拉格朗日反演，这是

$$\frac{(n-1)!}{(n-k-1)!}[x^k]\left(\frac{x}{\ln(1+x)}\right)^n$$

S1 同理：

$$\frac{(n-1)!}{(n-k-1)!}[x^k]\left(\frac{x}{1-e^{-x}}\right)^n$$

例 (SDOI D2T3，EI)

链接

$a_i = 1$ 大家都会做，这就是卡特兰数。
现在和标准的凸多边形三角剖分有什么区别呢？首先，同一条边上的点互相不能连边；其次，一些点可以不参与三角剖分。
我们先来扔掉第二个限制，这样就变成一些边不能连的三角剖分。
考虑容斥，强制连上这些边的一部分。首先连边肯定不能交叉；其次钦定的连边最多只能跨过一个点，否则其内部容斥结果为 0。
多边形不同的边是独立容斥的，所以我们算出每条边关于留下的边数的容斥 GF，分治 NTT 乘起来即可。
对于一条分成 m 段的边，我们的 GF 应该是

$$f_m(t) = [x^m]\frac{1}{1 - t(x - x^2)}$$

现在加上第二个限制，只需要枚举留下了多少个点：

$$\sum_{i=0}^{a-1} \binom{a-1}{i} f_{i+1}(t)$$

整理一下，这是

$$
\begin{aligned}
&\sum_{i=0}^{a-1}\binom{a-1}{a-1-i}[x^{i+1}]\frac{1}{1-t(x-x^2)}\\
=&[x^a]\frac{(1+x)^{a-1}}{1-t(x-x^2)}
\end{aligned}
$$

$x-x^2$ 的复合逆是 $\dfrac{1-\sqrt{1-4x}}{2}=xB_2(x)$

大概要算一个 $(1+xB_2)^{a-1}B_2^{-a-1}B_2'$，可以 $O(n)$.jpg，不过不是瓶颈

我们来换个方式推，直接在写 GF 的时候就考虑第二个限制。那么变成：每次从一个被选中的点开始，如果钦定一条边，那么枚举其连接到哪个点，然后在中间选择一个跨越的点；否则，就枚举下一个选择的点。这是

$$
\begin{aligned}
&[x^a]\frac{1}{1-t(-\sum_{i\geq 2}(i-1)x^i+\sum_{i\geq 1}x^i)} \\
=&[x^a]\frac{1}{1-t(\frac{x(1-2x)}{(1-x)^2})}
\end{aligned}
$$

同样是拉格朗日反演

例 (抽卡)

链接

假设停止后也能继续抽卡，但是不计贡献。这样对于每个 i，所有拥有 i 张卡的所有局面概率均等。我们计算出还未停止的概率，乘以转移到 $i+1$ 的期望时间，对所有 i 求和即得到答案。
现在问题转化为计算抽了 i 张卡还未停止的方案数。对于输入的 a_i，我们先将其划分为若干极长连续段，这样各连续段之间独立。我们对每个连续段计算出关于抽卡数量的 OGF，分治 NTT 乘起来即可。
现在考虑长度为 n 的连续段的 GF。还未停止的局面即若干长度小于 k 的极长连续段的拼接，这是

$$[x^{l+1}]\frac{1}{1-x\frac{1-(xt)^k}{1-xt}}$$

换个元就是

$$t^{l+1}[u^{l+1}]\frac{1}{1-t^{-1}\frac{u-u^{k+1}}{1-u}}$$

t^{-1} 和 t 处理起来差不多，后面就当成 t 了。

显然可以拉格朗日反演，但是复合逆怎么求？我们需要解方程

$$\frac{F - F^{k+1}}{1 - F} = x$$

牛顿迭代即可。
也可以不用拉格朗日反演，直接展开：

$$\begin{aligned}
&[x^n]\frac{1-x}{1-x-tx+tx^K}\\
=&[x^n]\frac{1-x}{1-x-tx}\frac{1}{1+\frac{tx^K}{1-x-tx}}\\
=&[x^n]\sum_{i\geq 0}(1-x)(-1)^i(tx^K)^i\frac{1}{(1-x-tx)^{i+1}}
\end{aligned}$$

前面的 $1-x$ 无关紧要，丢掉继续展开：

$$\sum_{iK\le n}(-t)^i[x^{n-iK}]\frac{1}{(1-x(1+t))^{i+1}}$$
$$=\sum_{iK\le n}(-t)^i(1+t)^{n-iK}\binom{n-iK+i}{i}$$

可以直接分治 NTT。实践上换元成 $1-t$ 会快很多。

q-analog

水很深，OI 把握不住！事实上这种东西有很深的数论背景。

例 (Jacobi 三重积，大概 OI 无关)

$$\prod_{r=1}^{\infty}(1-q^{2r})(1+q^{2r-1}z)(1+q^{2r-1}z^{-1}) = \sum_{k=-\infty}^{\infty} q^{k^2} z^k$$

q-二项式定理

我们来试着算一算

$$F(q,z)=\prod_{i=0}^{n-1}(1+q^iz)$$

关于 z 的展开式。做扰动:

$$F(q,qz)=\prod_{i=0}^{n-1}(1+q^{i+1}z)=F(q,z)\frac{1+q^nz}{1+z}$$

这得到了一个方程，把 $1+z$ 乘到左边并按 z 展开得到

$$f_i(q)=\frac{q^{i-1}-q^n}{1-q^i}f_{i-1}(q)$$

递归到 $f_0(q)$ 得到

$$f_i(q) = q^{i(i-1)/2}\frac{(1-q^{n-i+1})\cdots(1-q^n)}{(1-q^i)\cdots(1-q)}$$

换个形式:

$$f_i(q) = q^{i(i-1)/2}\frac{(1-q^n)\cdots(1-q)}{(1-q^i)\cdots(1-q)(1-q^{n-i}\cdots(1-q))}$$

注意形式 $(1-q^k)\cdots(1-q)$。记 $[k]!_q = \dfrac{(1-q^k)\cdots(1-q)}{(1-q)^k}$，称为 q-阶乘。这样我们应该定义 $[k]_q = \frac{1-q^k}{1-q} = \sum_{i=0}^{k-1} q^i$ 为 q-整数。右边就可以认为是 q-二项式系数 $\begin{pmatrix} n \\ i \end{pmatrix}_q$。这样我们就得到 q-二项式定理:

$$\prod_{i=0}^{n-1}(1+q^i z) = \sum_{i=0}^{n} q^{i(i-1)/2}\begin{pmatrix} n \\ i \end{pmatrix}_q z^i$$

q-二项式系数有很多和二项式系数相似的特性，有兴趣可以自己推一推。

二项式定理负指数的形式，其 q-模拟也有：

$$\prod_{i=0}^{n-1} \frac{1}{1-q^i z} = \sum_{k\geq 0} \binom{k+n-1}{k}_q z^k$$

单独拎出来证明的话和正指数差不多，或者你也可以把二项式系数的结论对着翻译到 q-形式然后立得

在上面两个式子里令 $n \to \infty$ 得到恒等式

$$\prod_{i\geq 0}(1+q^i z) = \sum_{i\geq 0} q^{i(i-1)/2} \frac{z^i}{(1-q^i)\cdots(1-q)}$$

$$\prod_{i\geq 0} \frac{1}{1-q^i z} = \sum_{i\geq 0} \frac{z^i}{(1-q^i)\cdots(1-q)}$$

逆序对

我们知道长为 n 的排列关于逆序对数量的 OGF 为

$$\prod_{i=1}^{n}\frac{1-q^i}{1-q}=[n]!_q$$

考虑 $[n]!_q$ 截取前 k 项怎么算，大概有以下几种方法：

- Ln-Exp，复杂度 $O(k\log k)$，需要多项式操作
- 如果 $n\geq k$，那么有五边形数定理（顺带一提，这是之前提到的 Jacobi 三重积的一个推论）

$$\prod_{i\geq 1}(1-z^i)=\sum_{i=-\infty}^{\infty}(-1)^i z^{(3i^2+i)/2}$$

  这只有 $O(\sqrt{k})$ 项，可以暴力和分母卷积。
- 组合意义，我不会做，不过应该有做法，请会的同学讲一下

- 按照 q-二项式定理展开分子:

$$\prod_{i=0}^{n-1}(1-q^i\cdot q)=\sum_{i\geq 0}(-1)^i q^{i(i+1)/2}\binom{n}{i}_q$$

注意我们可以在 $O(k)$ 内从 $\binom{n}{i}_q/(1-q)^n$ 递推到 $\binom{n}{i+1}_q/(1-q)^n$，而上式显然只有 $O(\sqrt{k})$ 项有贡献，于是我们得到了一个 $O(k\sqrt{k})$ 的做法，这个做法跑得比较快

拆分数

$$\prod_{i\geq 1}\frac{1}{1-z^i}$$

Ln-Exp 以外的标准做法是五边形数定理，因为其逆的项数为 $O(\sqrt{n})$。

研究拆分数的一个重要方法是 Ferrers diagram，对于一个拆分 $n = a_1 + \cdots + a_k, a_1 \leq \cdots \leq a_k$，我们在第 i 行画 a_i 个格子，就得到一张图表。

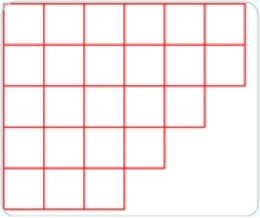

图: Ferrers diagram

通过图表立刻得到项数和值域是对称的。其导出另一种 $O(n\sqrt{n})$ 算法:

$$\prod_{i\geq 1}\frac{1}{1-z^i}=\sum_{i\geq 1}z^{i^2}\left(\prod_{k=1}^{i}\frac{1}{1-z^i}\right)^2$$

其组合意义是作直线 $y=x$，其与 Ferrers diagram 交于一点，以原点和此点为轴作正方形，则剩下的部分是两个值域 $<=i$ 的拆分数

例

项数不超过 n，值域不超过 m 的划分数

$$[t^n]\frac{1}{1-t}\prod_{i=1}^{m}\frac{1}{1-q^i t}=\binom{n+m}{n}_q$$

例

项数/值域不超过 n 的划分数

$$\prod_{i=1}^{n}\frac{1}{1-z^i}$$

如何快速计算？

对任何给定的 $A(z)$，$(1-z^k)\cdots(1-z)A(z)$ 可以在 $O(n\sqrt{n})$ 内计算前 n 项:

$$(1-z^k)\cdots(1-z)=\sum_{i\geq 0} z^{i(i+1)/2}(-1)^i\binom{k}{i}_z$$

只有 $O(\sqrt{n})$ 项有贡献，$\binom{k}{i}_z A(z)\to\binom{k}{i+1}_z A(z)$ 可以 $O(n)$

那么

$$\frac{A(z)}{(1-z^k)\cdots(1-z)}$$

也可以吗？

根号分治。或者，$\dfrac{A(z)}{\prod_{i\geq 1}(1-z^i)}$ 可以 $O(n\sqrt{n})$ 计算。

例 (Lucas)

链接

$$f(v) = [x^v y^m] \prod_{i=1}^{n} (1 + x^i y)$$

记 $B = 19190506$，则答案为

$$[y^m] B^p \prod_{i=1}^{n} (1 + B^{-i} y) = B^{p - m(m+1)/2} \binom{n}{m}_{B^{-1}}$$

关键在于求 $\binom{n}{m}_q$。

记 $a = ord(q)$ 为满足 $q^a \equiv 1 \pmod p$ 的最小整数，即 q 在模 p 下的阶。

仿照 Lucas 定理的证明，我们试着找一个 q-Lucas 定理。首先

$$\binom{a}{m}_q \equiv 0 \pmod{p}, 0 < m < a$$

直接展开分式即证。那么立刻得到

$$\prod_{i=0}^{a-1}(1+q^i z) \equiv 1 + z^a \pmod{p}$$

现在对 n 做 a 进制分解就得到

$$\begin{aligned}
&[z^m]\prod_{i=0}^{n-1}(1+q^i z)\\
\equiv&\left(\prod_{i=0}^{a-1}(1+q^i z)\right)^{\lfloor n/a \rfloor}\left(\prod_{i=0}^{n \bmod a-1}(1+q^i z)\right)\\
\equiv&\left([z^{\lfloor m/a \rfloor}](1+z)^{\lfloor n/a \rfloor}\right)\left([z^{m \bmod a}]\prod_{i=0}^{n \bmod a-1}(1+q^i z)\right)\\
\equiv&\binom{\lfloor n/a \rfloor}{\lfloor m/a \rfloor}\left([z^{m \bmod a}]\prod^{n \bmod a-1}(1+q^i z)\right)
\end{aligned}$$

注意展开的话不能两边消去 $q^{m(m-1)/2}$，因为 a 是偶数时不一定有 $C(m,2) \equiv C(m \bmod a, 2) \pmod a$

回到原题，你发现 $ord(B)$ 不大，所以预处理 $< a$ 的 q-阶乘即可。

$ord(B)$ 大的话能不能做呢？

快速 q-阶乘

先来回忆快速阶乘，为了方便，不妨设 $n = B^2$。我们计算

$$\prod_{i=1}^{B}(x+i)$$

在 $0, B, \cdots, n-B$ 处的点值，乘起来就得到 $n!$。显然可以多点求值，调块大小得到一个 $O(\sqrt{n}\log^{1.5} n)$ 的做法。然而多点求值的常数和代码复杂度都比较大，我们换个思路。考虑维护

$$F_d(x)\prod_{i=1}^{d}(x+i)$$

在 $0, B, \cdots, dB$ 处的点值，显然这唯一确定了这个多项式。我们将 d 从 1 倍增到 B。需要实现 $d+=1$ 和 $d*=2$。

- $d+=1$ 每个点值要乘一项，还要额外算一个 dB 的点值，总共 $O(d)$
- $d*=2$ 这是比较精妙的地方。我们分两步走，首先将点值范围倍增，再将多项式倍增
  - 点值倍增相当于实现点值平移 $F_d(iB+dB)$。记 $a_i=F_d(iB)$，$b_i=F_d(iB+\Delta)$，考虑拉格朗日插值

$$b_i=\sum_{k=0}^{d}a_k\prod_{j\neq k}\frac{(i-j)B+\Delta}{(k-j)B}$$

    观察后面这个乘积，分母只和 k 有关，可以预处理；分子是 $(\prod_{j=0}^{d}((i-j)B+\Delta)/((i-k)B+\Delta)$，前者只和 i 有关，后者只和 $i-k$ 有关。那么我们立刻知道这是个卷积（不过要把 b_i 平移一段），可以在 $O(M(2d))$ 时间计算
  - 多项式倍增注意 $F_{2d}(x)=F_d(x)F_d(x+d)$，因此也是点值平移

于是可以在 $O(\sqrt{n}\log n)$ 时间内求出 $n!$
多组询问怎么办？把 B 调小点，最后实现一个 N/B 长度的多项式平移，复杂度 $O(TB+B\log B+(N/B)\log(N/B))$。或者，二进制拆分 + 多点求值

现在回到 q-阶乘，思路是一致的，我们考虑

$$F_d(x) = \prod_{i=1}^{d}(1 - q^i x)$$

维护 $q^0, q^B, \cdots, q^{dB}$ 处的点值。$d+=1$ 依然平凡，考虑 $d*=2$。点值倍增相当于做 $F_d(q^{dB}q^{iB})$，多项式倍增相当于求 $F_d(q^d q^{iB})$。于是我们实现一个 $F_d(q^\Delta q^{iB})$ 的算法。此时拉格朗日插值变为

$$b_i = \sum_{k=0}^{d} a_k \prod_{j \neq k} \frac{q^{(i-j)B+\Delta} - 1}{q^{(k-j)B} - 1}$$

分母是依然只和 k 有关；分子是

$$\frac{\prod_j (q^{(i-j)B+\Delta} - 1)}{q^{(i-k)B+\Delta} - 1}$$

上面可以预处理，下面是只和 $i-k$ 有关的项，因此仍然可以卷积。由于始终有 $d < B$，所以这个除法总可以做。

另解

我们直接做这个多点求值。注意点值为 r^k 的多点求值是容易的：

$$f_k = \sum_i a_i r^{ik} = \sum_i a_i r^{\binom{i+k}{2}-\binom{i}{2}-\binom{k}{2}}$$

这是减法卷积。而 $F_d(x)$ 的系数容易计算，这样我们得到一个更简单的 $O(\sqrt{n}\log n)$ 做法。
我们可以看出这种递推和整式递推是类似的。对于一般情况，转移是矩阵形式

$$\mathbf{v}_i = Q(q^i)^{-1}M(q^i)\mathbf{v}_{i-1}$$

其中 $M(x)$ 是多项式矩阵，$Q(x)$ 是多项式。只要计算

$$M(q^B x)\cdots M(qx), Q(q^B x)\cdots Q(qx)$$

的点值即可。这个多项式矩阵的系数可以通过前面的扰动方法求得，于是两种方法都适用。

例 (哈希杀手，EI)

链接

第 0 步应该是拉格朗日插值:

$$f(x)=\sum_{i=0}^{n-1} f(q^i)\frac{\prod_{j\neq i}(x-q^j)}{\prod_{j\neq i}(q^i-q^j)}$$

首先考虑计算 k 个 $\prod_{j\neq i}(q^i-q^j)$，提出 q^i 后差不多是两个 q-阶乘，可以用前面说的方法计算。现在要算

$$f(x)=\left(\sum_{i=0}^{n-1}\frac{h_i}{x-q^i}\right)\prod_{i=0}^{n-1}(x-q^i)$$

先翻转系数

$$f^{rev}(x)=\left(\sum_{i=0}^{n-1}\frac{h_i}{1-q^i x}\right)\prod_{i=0}^{n-1}(1-q^i x)$$

然后是重头戏，我们怎么提取 $[x^m]f^{rev}(x)$ 呢？EI 展示了一个 递推方法，但是计算繁琐。我们来试着找一个更 q-binomial 一点的推法。

第 i 项的贡献是

$$\begin{aligned} f_i =& [x^m]\frac{1}{1-q^i x}\sum_j (-1)^j x^j q^{j(j-1)/2}\binom{n}{j}_q \\ =& \sum_{k=0}^{m} (-1)^k q^{k(k-1)/2}\binom{n}{k}_q q^{i(m-k)} \end{aligned}$$

这差不多是一个 q-二项式的截断。没有 q-analog 的时候我们推这种东西往往需要利用吸收恒等式（根据上下指标是否-1 可以得到三种，这里列出一种）

$$\frac{r-k+1}{k}\binom{r}{k-1} = \binom{r}{k}$$

容易验证它们的 q-analog 形式

$$\frac{1-q^{n-m+1}}{1-q^m}\binom{n}{m-1}_q = \binom{n}{m}_q$$

我们将基于此来得到 f_i 的递推式。

$$
\begin{aligned}
f_i =& q^{im}\sum_{k=0}^{m}(-1)^k q^{\binom{k}{2}-ik}\binom{n}{k}_q \\
=& q^{im}\sum_{k=0}^{m}(-1)^k q^{\binom{k}{2}-ik}((1-q^k)+q^k)\binom{n}{k}_q \\
=& q^{im}\sum_{k=1}^{m}(-1)^k q^{\binom{k}{2}-ik}(1-q^{n-k+1})\binom{n}{k-1}_q + q^m f_{i-1} \\
=& q^m f_{i-1} - q^{im}\sum_{k=0}^{m-1}(-1)^k q^{\binom{k+1}{2}-i(k+1)}(1-q^{n-k})\binom{n}{k}_q \\
=& q^m f_{i-1} + c_{i,m} - q^{im-i}\sum_{k=0}^{m}(-1)^k q^{\binom{k}{2}-ik}(q^k-q^n)\binom{n}{k}_q \\
=& q^m f_{i-1} + c_{i,m} + q^{n-i} f_i - q^{m-i} f_{i-1}
\end{aligned}
$$

这就得到

$$(1-q^{n-i})f_i = (q^m - q^{m-i})f_{i-1} + (-1)^m q^{\binom{m+1}{2}-i}(1-q^{n-m})\binom{n}{m}_q$$

即

$$(q^i - q^n)f_i = (q^{m+i} - q^m)f_{i-1} + (-1)^m q^{\binom{m+1}{2}}(1-q^{n-m})\binom{n}{m}_q$$

这是非齐次的 q-整式递推，可以直接上通解（写为向量 $\begin{bmatrix} f_i \\ 1 \end{bmatrix}$）。

例 (奇怪的题，dcx)

链接

题面有点巨大复杂（尊重原题），简化为：

$$F(x) = [t^n] \prod_{i=0}^{X-1} \frac{1-(tx^i)^K}{1-tx^i}$$

求 $[x^s]F/(1-x)$ 和 $[x^s]xF'/(1-x)$

$K \le n \le 50$，其他变量 $\le 10^9$

我们来看看 F 应该是什么样子，设 $Q(x,t)=\prod_{i=0}^{X-1}\dfrac{1-(tx^i)^K}{1-tx^i}$，那么 $Q(x,tx)=Q(x,t)\dfrac{1-(tx^X)^K}{1-tx^X}\dfrac{1-t}{1-t^K}$。基于此可以得到 $[t^n]Q$ 的一个递推式，记 $y=x^X$，则递推式即

$$(1-x^n)f_n(x)=(1-x^{n-1}y)f_{n-1}(x)+(y^K-x^{n-K})f_{n-K}(x)+(yx^{n-K-1}-y$$

那么容易看出 $f_n(x)=\dfrac{P_n(x,y)}{(1-x)\cdots(1-x^n)}$，且 $P_n(x,y)$ 关于 x,y 的次数分别为 $\Theta(n^2)$ 和 $\Theta(n)$。
暴力维护 $P_n(x,y)$ 即可做到 $O(n^5)$，用 FFT 可以 $O(n^4\log n)$。
求出 $f_n(x)$ 后是 $\Theta(n)$ 个线性递推，可以 $O(nM(n^2)\log s)$ 计算。
问题：还能做得更好吗？

例 (山河重整，EI)

给定整数集合 $S = \{1, \cdots, n\}$，计算有多少个子集 $T \subseteq S$，使得 $1, 2, \cdots, n$ 都可以被表示为 T 的一个子集中所有数的和。
$n \leq 5 \times 10^5$，任意模

观察一下 T 最小不能被表示的数的求法，我们发现如果 $k+1$ 是最小不能被表示的数，那么 T 必然存在一个子集，其和为 k 且能表示出 $1,\cdots,k$ 中的所有。我们记和为 k 且能表示出 $1,\cdots,k$ 的集合数量为 a_{k+1}，那么枚举最小的不能表示的数就得到方程

$$\sum_k a_k x^k (1+x^{k+1})(1+x^{k+2})\cdots = x(1+x)(1+x^2)\cdots$$

答案就是 $2^n - \sum_{k\le n} a_k 2^{n-k}$。左边就是

$$\begin{aligned}
&\sum_k a_k x^k (1+x^{k+1})\cdots \\
=&\sum_k a_k x^k \sum_i \frac{x^{i(k+1)} x^{i(i-1)/2}}{(1-x)\cdots(1-x^i)} \\
=&\sum_i \frac{x^{i(i+1)/2}}{(1-x)\cdots(1-x^i)} A(x^{i+1})
\end{aligned}$$

移项得到

$$A(x) = x(1+x)\cdots - \sum_{i\geq 1} \frac{x^{i(i+1)/2}}{(1-x)\cdots(1-x^i)} A(x^{i+1})$$

这样可以倍增计算。右边的和式只有 $O(\sqrt{n})$ 项有贡献，可以计算为

$$\left(\frac{x^{B(B+1)/2} A(x^{B+1})}{1-x^B} + x^{B(B-1)/2} A(x^B) \right) \frac{1}{1-x^{B-1}} + \cdots$$

于是可以在 $O(n\sqrt{n})$ 时间内计算。
更好的复杂度见 EI 博客

转置原理

转置原理给出一个用 $\mathbf{a} = A\mathbf{b}$ 的算法计算 $\mathbf{b} = A^T\mathbf{a}$ 的办法。注意前后的 $\mathbf{a}$ 和 $\mathbf{b}$ 没有关系。
如果问题可以被描述为计算线性变换 $\mathbf{a} = A\mathbf{b}$，其中 A 为常量，且算法过程中不会出现 b 的非一次项，那么这就是一个线性算法。我们可以看到相当多的问题都可以被这样改写：

例 (多项式乘法)

对于固定的 $C(x)$ 和输入的 $F(x)$，计算 $C(x)F(x)$。$A_{i+j,j} = C_i$

例 (FFT)

$A_{ij} = \omega^{ij}$。注意这是对称矩阵。

例 (复合)

对于固定的 $C(x)$ 和输入的 $F(x)$，计算 $F(C(x))$。$A_{ij} = [x^i]C^j$

线性算法在计算 $A\mathbf{b}$ 的时候通过分解 A 来完成，即计算 $A_1 \cdots A_m \mathbf{b}$。为了计算 $A^T\mathbf{b}$，只需要计算 $A_m^T \cdots A_1^T \mathbf{b}$。那么我们需要解决的问题是，分解 A，将其反向，然后逐个计算线性变换 $A_i^T \mathbf{v}$。这称为原算法的转置算法。

分解 A

为了解决这个问题，我们需要先分析一下算法的流程。首先，所有的条件控制都不应当与 $\mathbf{b}$ 有关。这样，分支结构和循环结构都可以被拆解为顺序结构。而顺序结构的每一条语句都应当是一个“直接的”线性变换，或者是定义常量。

例 (基本操作)

试着写出 $b_i+=cb_j, swap(b_i,b_j), b_i=b_j, b_i*=c, x=b_i, delete\ x$ 对应的变换矩阵，其中 c 为常量，x 为新定义的变量

$b_i=c, x=c$ 是合法的吗？它代表什么？

我们来特别说明一下函数，这是一个子算法，即它将 $\mathbf{b}$ 的一部分送进去（注意这个选一部分的操作也是线性变换），对其进行一些修改，再返回一个 $\mathbf{b}$ 的线性变换结果。返回这一步相当于创建了一些新的变量。

注意上面都是理论分析，实践上我们常常会把一部分操作压缩到一起（主要是临时变量的创建与删除），例如 FFT 的循环内有操作：$x=a_{j+k}, y=a_{i+j+k}*w[i+k]; a_{j+k}=x+y, a_{i+j+k}=x-y$，这就是 $\begin{bmatrix} a_{j+k} \\ a_{i+j+k} \end{bmatrix} \leftarrow \begin{bmatrix} 1 & \omega_{2i}^k \\ 1 & -\omega_{2i}^k \end{bmatrix} \begin{bmatrix} a_{j+k} \\ a_{i+j+k} \end{bmatrix}$

反向

反向的含义为颠倒自变量和因变量，然后倒序执行。我们还是来看看算法流程，对于循环结构，我们要把循环的顺序反转。注意这一步并不总是必须的，因为我们的循环未必具有前后依赖的关系（换句话说，这个循环对应的线性变换可以重排为分块对角阵）。例如 FFT 中里面两层循环都不需要反转，因为不同的 j, k 涉及到的 a 都不同。而最外层循环需要反转，它实际上模拟了一个递归分治的结构。
对于函数，我们要做什么？我们说过这是一个子算法，所以对其反转和前面一致，也即颠倒输入和返回值，然后逆序执行所有步骤。这正好能解决递归的问题。

$A^T\mathbf{b}$

只要 A 充分简单，这一步就是很容易的。

例 (基本操作的转置)

试着写出“分解 A”中基本操作的转置。

例 (FFT “蝴蝶变换”的转置)

试着写出“分解 A”中提到的 FFT 操作的转置

$$b_i+=cb_j \Longrightarrow b_j+=cb_i$$
$$swap(b_i, b_j) \Longrightarrow swap(b_i, b_j)$$
$$b_i = b_j \Longrightarrow b_j = b_i \qquad b_i* = c \Longrightarrow b_i* = c$$
$$x = b_i \Longrightarrow b_i = x$$
$$deletex \Longrightarrow x = 0$$

蝴蝶操作的转置为

$$\begin{bmatrix} a_{j+k} \\ a_{i+j+k} \end{bmatrix} \leftarrow \begin{bmatrix} 1 & 1 \\ \omega_{2i}^k & -\omega_{2i}^k \end{bmatrix} \begin{bmatrix} a_{j+k} \\ a_{i+j+k} \end{bmatrix}$$

这就是 $x = a_{j+k} + a_{i+j+k}, y = a_{j+k} - a_{i+j+k}; a_{j+k} = x, a_{i+j+k} = w[i+k] * y$

试看看!

我们来看一些应用。在使用转置原理时，要想清楚哪些是输入。

例 (区间加)

给定 m 个修改 (L_i, R_i, q_i)，对 $j \in [L_i, R_i]$ 的 a_j 加 q_i，最后输出所有 a_j。序列长度为 n。

我们来试试对这个问题应用转置原理！我们将 q_i 视作输入，a_i 视作输出，那么变换矩阵是 $A_{ij} = [L_j \le i \le R_j]$。
我们看看 $A^T\mathbf{a}$ 是什么，这是 $\sum_j [L_i \le j \le R_i] a_j$，即区间和。
想想区间和的做法：首先计算前缀和 $i = 2 \cdots n, a_i += a_{i-1}$，输出是 $q_i = a_{R_i} - a_{L_i - 1}$
再想想区间加的做法：差分为 $a_{L_i-1} -= q_i, a_{R_i} += q_i$，最后做后缀和 $i = n \cdots 2, a_{i-1} += a_i$
这刚好是一对互为转置的算法。

例 (DIF-DIT 实现的 FFT，据说很快)

试着将 DFT 的过程转置，而 IDFT 使用原算法（指使用 ω^{-1} 的做法）

附原算法卡常技巧：

- 单位根必须预处理，不然很慢（用复数的话可能为了精度要现算）
- 用 ull 存，加减法不取模，乘法取模（注意减法应该先 +mod）。此操作有风险，为了保险需要每若干轮取一次模

由于 FFT 的矩阵是对称的，所以转置算法的结果和原算法一致。
众所周知，FFT 分为两部分，一部分是 bitrev，一部分是迭代。
迭代的部分先前已经说过，而 bitrev 的转置还是 bitrev，不过要放到迭代之后。
现在和 IDFT 拼起来，因为 IDFT 开头也有个 bitrev，这样两个 bitrev 抵消，可以节省不少常数（大跨度内存访问带来的）
烫知识：写得比较好（可能处理单位根就行了）的 NTT 常数非常小，小于做 n 次指数为 n 的快速幂

多项式乘法的转置（MULT）

多项式乘法：对于固定的 $C(x)$ 和输入的 $F(x)$，计算 $C(x)F(x)$
截取前 n 项的结果
这个问题的转置是什么？

$A_{i+j,j} = C_i$，转置为 $A^T_{j,i+j} = C_i$，那么

$$ans_j = \sum_j C_i F_{i+j}$$

也就是“减法卷积”

多点求值

我们先来表述一下问题，即对于给定的 $n-1$ 次多项式 $F(x)=\sum_i a_i x^i$ 和 m 个点值 $x_0,\cdots,x_{m-1}$，计算所有 $F(x_i)$。也就是

$$q_i=\sum_j f_j x_i^j$$

显而易见，我们应该把 f 作为输入。那么 $A_{ij}=x_i^j$

传统的做法基于这样一个事实：$F(x_i) = F(x) \bmod (x - x_i)$。这样，我们分治计算出 $F(x) \bmod \prod_{i\in[L,R]}(x - x_i)$ 即可。
新的算法由转置原理得到。我们先来考察转置问题的计算，这是

$$f_i = \sum_j q_j x_j^i = [t^i] \sum_j \frac{q_j}{1 - x_j t}$$

那么我们通分之后分治计算这个东西，对每个节点 S 计算限定在这个节点对应区间上的多项式 $H_S(x)$。设 $U_S(x) = \prod_{i\in S}(1 - x_i x)$，两个子节点分别为 L, R，则此节点的返回值为 $H_L(x)U_R(x) + H_R(x)U_L(x)$，最后把根节点的返回值乘以 $U_{root}(x)^{-1}$。
现在我们将这个算法转置。首先要搞清楚哪些是常量，显然是所有 $U_S(x)$。所有 U 和 H 之间的乘法都应变为 MULT。转置后的流程为：先将输入（即 $F(x)$）和 $U_{root}(x)^{-1}$ 做 MULT，然后传入分治。对每个节点，将接受的输入 $\hat{H}_S(x)$ 和 $U_{R/L}(x)$ 做 MULT 传给 L/R，最后叶子就是答案。
skip2004 和 negiizhao 各自以这个为主体实现了一个 1s1e6 的多点求值（有巨大多优化，我也不懂），通过了 uoj500（这是 $x_i = r^i$ 的多点求值）

例 (Do Use FFT, GYM102978D)

给定长为 N 的序列 A, B, C，对 $k = 1, \cdots, N$ 求出

$$\sum_{i=1}^{n} C_i \prod_{j=1}^{k} (A_i + B_j)$$

答案对 998244353 取模，$N \leq 2.5 \times 10^5$

显然只有 C 是一次的，所以我们把 C 作为输入。这样变换矩阵就是

$$M_{ij} = \prod_{k=1}^{i} (A_j + B_k)$$

这个问题的转置是

$$C_i = \sum_{j=1}^{N} D_j \prod_{k=1}^{j} (A_i + B_k) = \sum_{j=1}^{N} D_j \prod_{k=1}^{j} (x + B_k) \Bigg|_{x=A_i}$$

也就是分治 NTT+ 多点求值，分别将这两部分转置即可。

例（「」）

链接

EI の神谕

让我们看看 EI 老师怎么说：

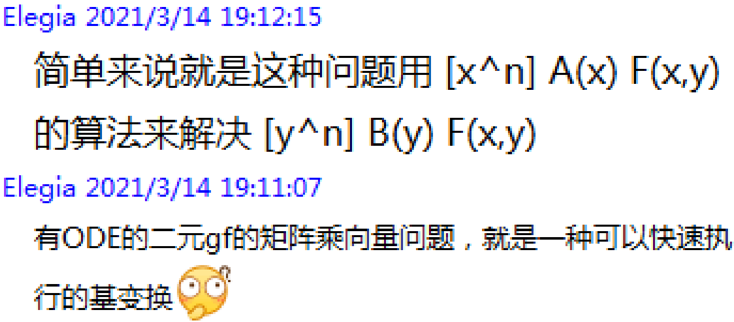

图: 你学会了吗

我们来解释一下这两句话。上面一句比较简单，它是说我们想求

$$ans_i = \sum_j b_j[x^i y^j]F(x,y)$$

其转置为

$$ans_j = \sum_i a_i[x^i y^j]F(x,y)$$

这就是第一句话的含义了。接下来我们看看第二句话，对输入的多项式 $A(y)$ 和 BGF $F(x,y)$，求 $[y^n]A(y)F(x,y)$，其中 $F(x,y)$ 有 y 方向的常数阶 ODE

$$\sum_{i=0}^{k} f_i(x,y)\frac{\partial^i}{\partial y^i}F = 0$$

按 y 展开 F 得到 $F(x,y) = \sum_n f_n(x)y^i$，于是上面的 ODE 给出了 $f_n(x)$ 的一个递推式。我们用矩阵形式写这个递推，则所求即

$$\sum_i a_i[1\ 0\ \cdots\ 0]M_i(x)M_{i-1}(x)\cdots M_t(x)\mathbf{v}_0(x)$$

显然可以分治计算，复杂度 $O(n\log^2 n)$。

回到这个题，我们先看看长度为 m 且有 i 个环的方案数，这应该是

$$\frac{m!}{i!}[x^m]\left(\ln\left(\frac{1}{1-x}\right)-x\right)^i$$

那么我们要算的就是

$$h_m = m![x^m]\sum_i \frac{g_i}{i!}\left(\ln\left(\frac{1}{1-x}\right)-x\right)^i$$

也就是 $[y^n]G(y)\exp\left(y\left(\ln\left(\frac{1}{1-x}\right)-x\right)\right)$。（为什么是 exp？$1/(1-yF)$ 可以吗？)
我们注意后面这个 BGF 有 x 方向的 ODE：

$$\frac{\partial F}{\partial x} = F\cdot\frac{xy}{1-x}$$

这样我们就转化为了上一页的问题。

例 (SDOI D2T3，但是转置原理)

计算 $[x^n]\dfrac{A(x)}{1-tx(1-x)}$

转置问题应该是复合 $x(1-x)$。
我们知道复合一次多项式可以 $O(n\log n)$，事实上二次也可以。
配方：

$$ax^2+bx+c=a(x+\frac{b}{2a})^2+\frac{4ac-b^2}{4a}$$

这是 $(Ax+C)\circ x^2\circ(x+B)$
Polynomial,djq 给出了一个能 $O(n\log n)$ 复合的多项式族，以及判定和拆复合链的方法

另外两个基本复合

复合 e^x-1 和 $\ln(1+x)$ 可以在 $O(n\log^2 n)$ 时间内完成。对于前者，考虑 $F(e^x)$ 即可，这是

$$[x^i]F(e^x)=\frac{1}{i!}\sum_j f_j j^i=\frac{1}{i!}[x^i]\sum_j\frac{f_j}{1-jx}$$

后者 rushcheyo 给了个有点麻烦的做法，可以简单一些，我们考虑复合 $\ln(1-x)$，这是

$$\begin{aligned}[x^i]F(\ln(1-x))&=\sum_j f_j(-1)^j[x^i]\ln\left(\frac{1}{1-x}\right)^j\\&=\frac{1}{i!}\sum_j f_j(-1)^j j!\begin{bmatrix}i\\j\end{bmatrix}\end{aligned}$$

其转置为

$$ans_j=\sum_i g_i\begin{bmatrix}i\\j\end{bmatrix}=[x^j]\sum_i g_i x^{\overline{i}}$$

可以分治，也许本质上是一样的。更多的结果可以参考 2020 集训队论文《转置原理及其应用》

容斥

直接来看点例子吧

例 (计树，EternalAlexander)

链接

注意到最终的树由若干条 $l, l+1, \cdots, r(r-l \geq 1)$ 的极长链拼接而成。如果没有极长的限制，我们利用 prufer 序列的结论：将大小为 $a_1, \cdots, a_m$ 的给定树拼成一棵树的方案数为

$$n^{m-2} \prod a_i$$

即可写出答案的表达式：

$$n^{-2}[x^n] \frac{1}{1 - n \sum_{i \geq 2} i x^i}$$

现在考虑容斥掉极长的限制，我们在长度为 i 的链前面配上系数 f_i，然后依然像上面那样写表达式。那么关键在于 f_i 要取得合适，使得系数刚好凑成我们需要的。注意现在一个最终形成的树的方案的系数为所有拼成它的方法的 f_i 乘积之和。我们对每个极长段凑出这个系数：

$$\frac{1}{1-F} - 1 = \frac{x^2}{1-x}$$

那么 $F(x) = \frac{x^2}{1-x+x^2}$

现在可以写出答案的表达式:

$$n^{-2}[x^n]\frac{1}{1-n\sum_i if_ix^i}$$

这是个线性递推:

$$n^{-2}[x^n]\frac{1}{1-n(xF)'}$$

于是此题可以 $O(\log n)$ 完成。

例 (jiangly 的排列数数题)

问对于所有长为 n 的排列，有多少排列存在一个连续上升段 $\geq k$。对所有 k 回答，对大质数取模。

首先容斥为不存在，那么和前一题一样，我们容斥掉所有极长段。这时的容斥系数是

$$\frac{1}{1-F}-1=\sum_{1\le i<k}x^i=\frac{x-x^k}{1-x}$$

然后将带容斥系数的段拼起来，这是

$$n![x^n]\frac{1}{1-\sum_i f_i x^i/i!}$$

注意 $F(x)=\dfrac{x-x^k}{1-x^k}=\sum_{i\ge 0}x^{ik+1}-x^{ik+k}$ 只有 $O(n/k)$ 项，所以这个求逆暴力计算是 $O(n^2/k)$ 的，总共 $O(n^2\log n)$。进一步的优化参考 EI 博客，不是重点就不展开了。

例 (U 群把妹王，EI)

链接

题意大概是说，行和列的图案必须形成若干个大小在 S/T 中的等价类。我们将此限制容斥掉，对每种大小的等价类赋予一个系数 a_i 后进行 (EGF) 拼接。以 S 为例，我们应该凑成

$$\exp(\sum_i a_i x^i/i!) - 1 = \sum_{i\in S} \frac{x^i}{i!}$$

那么现在我们计算出把 n 行分为 i 个伪等价类（等价类之间还可能合并，我们用容斥系数去掉了此限制）的带系数方案数，即

$$f_i = n![x^n]\frac{A^i}{i!} = \frac{n!}{i!}[x^n]\left(\ln\left(1+\sum_{i\in S}\frac{x^i}{i!}\right)\right)^i$$

列的计算也类似，记为 g_i，那么答案就是

$$\sum_{i,j} f_i g_j c^{ij}$$

由经典的 $c^{ij} = c^{\binom{i+j}{2}-\binom{i}{2}-\binom{j}{2}}$ 即可 $O(n\log n)$ 计算。

还需要解决所有 f_i 的计算，这是 $\exp(u\ln(1+S(x)))$，或者 $\dfrac{1}{1-u\ln(1+S(x))}$ 也行。由于 $S(x)$ 项数很少，故 $\ln(1+S(x))$ 的复合逆可以通过牛顿迭代在 $O(an\log n)$ 时间内计算。这样可以通过拉格朗日反演计算所有 f_i

反射容斥

从 $(0,0)$ 走到 (n,m)，每步的向量是 $(1,\pm 1)$，不允许碰到 $y=A$ 和 $Y=B$ 两条线（$A<0<B$）

只有 $y=B$ 的话，答案就是无限制走到 (n,m) 的方案数减掉无限制走到 $(n,2B-m)$ 的方案数，前提是 $m<B$，否则是 0。两个组合数减一下即可。
现在加上 $y=A$。我们的结论是：总的-碰到 A 的-碰到 B 的 $+$ 碰到 AB 的 $+$ 碰到 BA 的-碰到 ABA 的-碰到 BAB 的$\cdots\cdots\cdots\cdots$
这里“碰到”只计入从起点或者另一边回来的第一次。碰到 $ABAB$ 的方案数如下计算：将 (n,m) 沿着 $BABA$ 翻折，然后计算从 $(0,0)$ 走到此点的方案数。其他也类似。
为什么是对的？一个碰了奇数次的方案会被计入 $1-2+2-2+2\cdots-1=0$ 次，碰了偶数次的方案会被计入 $1-2+2-2+2\cdots-2+1$ 次。
注意沿着 AB 反射一次会使得 m 增加 $2(B-A)$，所以结果是 $O(n/(B-A))$ 个组合数。
也可以用解析方法得到，见 EI 博客

例 (count)

链接

首先 $n<m$ 无解。题意中的同构相当于笛卡尔树同构，那么我们直接数笛卡尔树就可以了。限制相当于左偏深度不超过 m，每一棵这样的树都可以还原出一个序列。有经典的二叉树到括号序列的同构：$seq(T)=(seq(T_L))seq(T_R)$，这样限制成为任何时刻前缀剩下的 “(” 个数不超过 m 的括号序列数量，这也就是从 $(0,0)$ 走 $2n$ 步 $(1,\pm 1)$ 到 $(2n,0)$ 不碰到 $y=-1$ 和 $y=m+1$ 的方案数，也就是反射容斥板子。
也可以生成函数，$F_m(x)=1+xF_{m-1}(x)F_m(x)$，即 $F_m=\dfrac{1}{1-xF_{m-1}}$。设 $F_m=\dfrac{A_m}{B_m}$，那么有转移

$$\begin{bmatrix}A_m\\B_m\end{bmatrix}=\begin{bmatrix}0&1\\-x&1\end{bmatrix}\begin{bmatrix}A_{m-1}\\B_{m-1}\end{bmatrix}=\begin{bmatrix}0&1\\-x&1\end{bmatrix}^m\begin{bmatrix}1\\1\end{bmatrix}$$

有很多做法可以算这个东西，比如暴力快速幂，或者代入所有单位根插值最后做一次 IDFT，或者直接推通项。前两者只能 $O(n\log n)$，推通项的做法见 EI 博客，也可以 $O(n)$

例 (百度搜索反射容斥找到的题)

用 n 个 1 和 m 个 0 构造序列使得任何子区间包含的 $0,1$ 个数相差不超过 k，计数。$n+m,k \leq 5 \times 10^7$

即从 $(0,0)$ 走 $(1,\pm1)$ 到 $(n+m,n-m)$ ，期间任何两点高度差不超过 k。只要最高点和最低点差不超过 k 即可，那么我们来枚举最高点高度 R，计算最高点恰为 R 且最低点不低于 $R-k$ 的方案数。再做一步容斥就是最高点不超过 R 的减去不超过 $R-1$ 的，那么单次方案数可以 $O((n+m)/k)$ 计算。注意起点为 $(0,0)$，所以要限制 $0 \le R \le k$，那么只要做 $O(k)$ 次计算，总共 $O(n+m)$

求导

注意对 x^k 求 r 次导会得到 $k^{\underline{r}}x^{k-r}$，而算子 $\vartheta = xD$ 作用 r 次会得到 $k^r x^k$。
有必要提一下 xD 的幂的运算，注意 $xD(x^iD^i) = ix^iD^i + x^{i+1}D^{i+1}$，通过归纳容易证明

$$(xD)^n = \sum \left\{ {n \atop i} \right\} x^i D^i$$

这和普通幂展开成下降幂的结果是一致的。此外有乘积的高阶导公式：

$$(fg)^{(n)} = \sum_i \binom{n}{i} f^{(i)} g^{(n-i)}$$

试看看

例 (联合省选 2020，组合数问题)

链接

我们把原题里的 x 记作 r，而将 x 作为形式元，所求即

$$\left(\sum_i a_i (xD)^i\right)(1+rx)^n\bigg|_{x=1}$$

我们可以通过转置原理计算前面的算子展开，转置后为

$$\sum_j \frac{a_j x^j}{(1-x)\cdots(1-jx)}$$

可以分治 NTT 计算。

例 (qyc 给的题)

$$\sum_{i=0}^{m}(-1)^i\binom{i}{r}\Big/\binom{n}{i}$$

注意 beta 积分

$$\int_0^1 x^n(1-x)^m\ \mathrm{d}x=\frac{n!m!}{(n+m+1)!}$$

你需要给一个 $O(r)$ 的做法。

按照给出的 beta 积分，我们可以将问题写作

$$(n+1)\int_0^1 \sum_{i=0}^{m} \binom{i}{r} (-x)^i (1-x)^{n-i} \, \mathrm{d}x$$

定义 $F(x,t)=\sum_{i=0}^{m} x^i t^{n-i} = t^{n-m}\dfrac{x^{m+1}-t^{m+1}}{x-t}$，那么有

$$\frac{x^r}{r!}\frac{\partial^r}{\partial x^r}F(x,t)=\sum_{i=0}^{m}\binom{i}{r}x^i t^{n-i}$$

代入 $x=-u, t=1-u$ 就得到我们要求的式子。另一方面我们直接对封闭形式求导得到

$$\frac{x^r}{r!}\frac{\partial^r}{\partial x^r}F(x,t)=\frac{x^r}{r!}\sum_{i}\binom{r}{i}t^{n-m}\left(x^{m+1}-t^{m+1}\right)^{(r-i)} i!(-1)^i(x-t)^{-i-1}$$

代入 $x=-u, t=1-u$ 之后得到 $O(r)$ 个 beta 积分。$(-1)^i$ 的存在刚好使得分母变成常数。

使用一个随机数生成器生成序列 $a_1, \cdots, a_n$，生成 i 的概率是 $\binom{M}{i} p^i (1-p)^{M-i}$。给一个不超过 N 次的多项式 f 在 $0, \cdots, N$ 处的点值。设

$$g(a_1, \cdots, a_n) = \sum_{0 \le b_i \le a_i} f(b_1 + \cdots + b_n)$$

求 g 的期望。
$N \le 10^5, n, M, p < 998244353$

所有 b 独立，设

$$G(x) = \sum_{b \geq 0} x^b \sum_{a \geq b} \binom{M}{a} p^a (1-p)^{M-a}$$

那么答案是

$$\sum_{i \geq 0} f(i)[x^i]G(x)^n$$

注意

$$\sum_{i \geq 0} i^{\underline{k}}[x^i]H(x) = H^{(k)}(1)$$

所以我们先把 f 转成下降幂基

$$
\begin{aligned}
&\sum_{i\geq 0}\frac{t^i}{i!}f(i)\\
=&\sum_{i\geq 0}\frac{t^i}{i!}\sum_j b_j i^{\underline{j}}\\
=&\sum_j b_j\sum_i\frac{t^i i^{\underline{j}}}{i!}\\
=&e^t\sum_j b_j t^j
\end{aligned}
$$

于是可以 $O(n\log n)$ 完成点值到下降幂基的转换

另一个问题是计算 $D^k(G^n)$，我们首先注意

$$D^k(fg) = k![t^k]\left(\sum_{i\geq 0}\frac{f^{(i)}t^i}{i!}\right)\left(\sum_{i\geq 0}\frac{g^{(i)}t^i}{i!}\right)$$

可以将其自然扩展到多个函数乘积的高阶导，即

$$D^k(G^n) = k![t^k]\left(\sum_{i\geq 0}\frac{G^{(i)}t^i}{i!}\right)^n$$

于是我们只要计算 $G^{(i)}(1)$。

G 的导数容易计算：

$$\begin{aligned}
G^{(k)}(1) &= \sum_{b\geq 0} b^{\underline{k}} \sum_{a\geq b} \binom{M}{a} p^a (1-p)^{M-a} \\
&= \sum_{a\geq 0} \binom{M}{a} p^a (1-p)^{M-a} \sum_{b=0}^{a} b^{\underline{k}} \\
&= \sum_{a\geq 0} \binom{M}{a} p^a (1-p)^{M-a} \frac{a^{\underline{k+1}}}{k+1} \\
&= \frac{k!}{k+1} \sum_{a\geq 0} \binom{M}{a} p^a (1-p)^{M-a} (a+1) \binom{a}{k} \\
&= \binom{M}{k} \frac{k!}{k+1} \sum_{a} \binom{M-k}{a-k} p^a (1-p)^{M-a} (a+1) \\
&= \binom{M}{k} \frac{k!}{k+1} p^a \sum_{a} \binom{M-k}{a} p^a (1-p)^{M-k-a} (a+(k+1)) \\
&= \binom{M}{k} \frac{k!}{k+1} p^a (k+1+p(M-k))
\end{aligned}$$

换元 $x = e^t$

注意 $\sum_i i^k [x^i] F(x) = k![t^k] F(e^t)$
所以这个办法也可以处理一些点积 i^k 的情况。常见的如“所有方案的（权值的和）或者（长度）的 k 次幂的和”。

例 (生成树计数)

链接

矩阵树可以算所有生成树边权乘积的和。我们把边权变成 e^{at} 即可。截断到 x^k。
求行列式要稍微注意一下，逆可能不存在，所以消元时选一个次数最低的消其他行或者辗转相除（不过我看好像没人处理这个，不知道为啥）

例（「美团 CodeM 决赛」bt）

给定 n, m，对所有 $0 \le k \le m$ 求

$$\sum_{T \text{is binary tree}, |T|=n} \sum_{u,v \in \text{leaf}(T), u \le v} \text{len}(u, v)^k$$

其中 $len(u, v)$ 是 $u - v$ 路径上的节点数量。

$n \le 10^7$，$m \le 300$

记 F, G, H 是树的数量，直上直下路径的统计，所有路径的统计，有

$$
\begin{aligned}
F &= x(1+F)^2 \\
G &= x(1+2(1+F)G)e^t \\
H &= x(e^t + 2(1+F)H + G^2 e^t)
\end{aligned}
$$

解出

$$
H = \frac{xe^t}{\sqrt{1-4x}} + \frac{x^3 e^{3t}}{\sqrt{1-4x}(1-(1-\sqrt{1-4x})e^t)^2}
$$

为了按照 t 一维展开，记 $y = e^t - 1, \lambda = \sqrt{1-4x}$，那么

$$
\begin{aligned}
H =& xe^t/\lambda + \frac{x^3(y+1)^3}{\lambda(1-(1-\lambda)(1+y))^2} \\
=& xe^t/\lambda + \frac{x^3(y+1)^3}{\lambda^3(1-y(\lambda^{-1}-1))^2}
\end{aligned}
$$

展开 $\dfrac{1}{1-y(\lambda^{-1}-1)}$ 即可。$[x^n]\lambda^{-k}$ 容易 $O(n)$ 预处理后 $O(1)$ 计算。最后还原回 t^i 基，这一步 $O(m^2)$ 简单完成即可。

单位根反演与 $\bmod (x^n - 1)$

$$[n|k] = \frac{1}{n}\sum_{i=0}^{n-1}\omega_n^{ik} = [x^0]\left(x^k \bmod (x^n - 1)\right)$$

右端等式就是 IDFT。一般右边多项式的方法威力更大，但是和式有时更简便

例 (生成树求和)

链接

位之间独立，变为 $O(\log c)$ 次计算：边权和为 $0,1,2$ 的方案数分别是多少。
把边权变成 x^v，然后矩阵树。在 $\bmod(x^3-1)$ 下做乘法。
求逆还是不好求。注意因式分解 $x^3-1=(x-1)(x^2-x+1)$，可以证明这两个多项式在 $\mathbb{F}_3$ 上不可约，于是可以 CRT 合并出结果。
对于一般的 x^n-1 的因式分解可以考虑分圆多项式：
$\Phi_n(x)=\prod_{\gcd(n,d)=1}(x-\omega_n^d)$，那么易见 $x^n-1=\prod_{d|n}\Phi_d(x)$，以及 $\Phi_n(x)=\prod_{d|n}(x^d-1)^{\mu(n/d)}$。这是整系数多项式，且它在 $\mathbb{Q}$ 上不可约。
$\mathbb{F}_p$ 上的 n 次多项式的不可约性检测方法是验证充要条件：
$f|(x^{p^n}-x)$ 且对任意素数 $t|n$，有 $\gcd(x^{p^{n/t}}-x,f)=1$。
有限域多项式的一些理论我不太懂（比如上面这个结论），所以只能处理一点最简单的情况。要学的话大概要整点代数教程来看（比如《代数学引论》，这书真的很好）

不展开讲的例子

下面是两个和分圆多项式相关的例子，有兴趣可以自己了解。

例 (算力训练)

算力训练

例 (复读机)

链接

高阶差分

定义差分算子 $\Delta f(x) = f(x+1) - f(x)$。定义移位算子 $Ef(x) = f(x+1)$，那么 $\Delta = E - 1$。由于 E 和 1 可交换，我们很容易确认高阶差分的表达式：

$$\Delta^n = \sum_i \binom{n}{i} E^i (-1)^{n-i}$$

作用到 f 上就是

$$\Delta^n f(x) = \sum_i \binom{n}{i} f(x+i)(-1)^{n-i}$$

试着写出 $x = 0$ 的表达式，这是最常见的情况。注意 n 次多项式差分之后会变成 $n-1$ 次多项式，基于这一点可以得出来一些看起来非常匪夷所思的结果。

注意 $\Delta^k x^{\underline{n}} = n^{\underline{k}} x^{\underline{n-k}}$，所以对于多项式 $f(x)$ 有

$$f(x) = \sum_{i \geq 0} \Delta^i f(0) \frac{x^{\underline{i}}}{i!}$$

通过把 f 变换到下降幂基很容易验证这一点。这是泰勒展开的一个离散模拟。

例

$$\sum_k \binom{n}{k} \binom{r - sk}{n} (-1)^k = s^n, f(k) = \binom{r - sk}{n}$$

如果把右边组合数的下指标改成 m 并设 $d = m - n$，你能得到一个 $O(d \log d)$ 的做法吗？

假如说我们展开了 $\binom{r-sx}{m}$，那么应该只有第 n 到 m 次项有贡献。我们首先求出这些项的系数，翻转系数得到

$$\frac{1}{m!}(rx-s)((r-1)x-s)\cdots((r-m+1)x-s) \bmod x^{m-n+1}$$

我们姑且忽略前面的阶乘，那么提出 $(-s)^m$ 再取对数，就是

$$-\sum_{i=0}^{m-1}\sum_{j\geq 1}\frac{x^j}{js^j}(r-i)^j$$

那么我们只需要求出两个自然数幂和即可。这大概是

$$\sum_{i\geq 0}\frac{x^i}{i!}\sum_{j=0}^{n-1}j^i=\frac{1-e^{nx}}{1-e^x}$$

容易 $O(d\log d)$ 计算。最后 exp 回去也是 $O(d\log d)$。

现在 x^{m-i} 的贡献是 $n!\left\{\begin{matrix} m-i \\ n \end{matrix}\right\}$，我们需要求一列斯特林数，即

$$\frac{(m-i)!}{n!}[x^{m-n-i}]\left(\frac{e^x-1}{x}\right)^n$$

容易 $O(d\log d)$ 计算，前面剩下的一些阶乘互相抵消，可以 $O(d)$ 算出所有系数。
bonus：在 $O(d\log^2 d)$ 时间内算出所有系数。通过转置原理应该不难得到。

例 (James Stirling，OI 无关)

$$\ln x! = \sum_{i\geq 0} s_i \binom{x}{i}$$

其中

$$\begin{aligned} s_n &= \Delta^n (\ln x!)|_{x=0} \\ &= \Delta^{n-1} \ln(x+1) \\ &= \sum_i \binom{n-1}{i} (-1)^{n-1-i} \ln(i+1) \end{aligned}$$

可以证明这个级数对 $x > -1$ 收敛，这样就得到阶乘在 $\mathbb{R}$ 上的一个延拓。这和欧拉积分定义的阶乘

$$x! = \int_0^{+\infty} e^{-t} t^x \, \mathrm{d}t$$

相等。

例 ($O(n^3)$ 算 0)

$$\sum_i \sum_j \binom{n}{i} \binom{n}{j} (-1)^{i+j} \int_0^{-i} x^{n-1}(x+i-j)^{n-1} \, \mathrm{d}x$$

由于积分限里有 i，把和 j 无关的拿到前面去，也就是要算

$$\sum_j \binom{n}{j}(-1)^j(x+i-j)^{n+1} = \sum_j \binom{n}{j}(-1)^{n-j}(x+i-n+j)^{n+1}$$

差分完变成一次多项式，所以我们手动计算这两个系数得到

$$(n+1)!(x+(i-\frac{n}{2}))$$

乘上前面的 x^{n-1} 积分出来又变成 i 的多项式，又是一个高阶差

分，最后算出来

$$0$$