浅谈多项式牛顿迭代与拉格朗日反演在 OI 中的应用

华东师范大学第二附属中学 左骏驰

摘 要

生成函数的计数问题是 OI 中的一类重要的问题。近年来，除了多项式乘法、多项式求逆、多项式求 exp、多项式求 ln 等传统方法，牛顿迭代、拉格朗日反演等新的技术层出不穷。

因此，本文将围绕着牛顿迭代与拉格朗日反演这两个主题，浅析它们在 OI 中的应用。本文将先介绍牛顿迭代、拉格朗日反演的基本知识，然后介绍它们在 OI 题目中的应用，并将这些用法进行归纳、分类。接着本文介绍了牛顿迭代法的高级应用。由于牛顿迭代和拉格朗日反演都和多项式复合、复合逆息息相关，本文还简要介绍了求多项式复合、复合逆的方法。

1 前置知识

对于任意一个域 F 和其上的任意一个 $n+1$ 项的序列 $a_0, a_1, \cdots, a_n \in F$，我们记这个序列的**普通型生成函数 (也称 OGF)** 为:

$$F(x) = \sum_{i=0}^{n} a_i x^i \tag{1}$$

其**指数型生成函数 (也称 EGF)** 为:

$$G(x) = \sum_{i=0}^{n} \frac{a_i}{i!} x^i \tag{2}$$

而对于无穷序列 $a_0, a_1, \cdots$ 仍然可以定义其普通型和指数型生成函数：

$$F(x) = \sum_{i=0}^{+\infty} a_i x^i, \tag{3}$$

$$G(x) = \sum_{i=0}^{+\infty} \frac{a_i}{i!} x^i \tag{4}$$

对任意一个形如如下形式的级数：

$$F(x) = \sum_{i=-\infty}^{+\infty} a_i x^i \tag{5}$$

满足 $a_{-1}, a_{-2}, \cdots$ 这个数列在足够多项之后为 0，则称 $F(x)$ 为形式 Laurent 级数。我们可以约定 $[x^n]F(x)$ **表示** a_n，类似地，我们也可以在其上定义加减法、乘法、求导等运算。但在之后的论述中，”函数”均指的是“形式幂级数”，也就是不存在 $x^-1, x^-2, \cdots$ 项的形式 Laurent 级数。

对于生成函数 $F(x), G(x)$ (下面均约定它没有 $x^{-1}, x^{-2}, \cdots$ 这些项)，若 $F(0) \neq 0, F(x)G(x) = 1$，则我们称 $G(x)$ 为 $F(x)$ 的逆，记为: $G(x) = \frac{1}{F(x)}$。

对于生成函数 $F(x), G(x)$, 若 $[x^0]F(0) = 0, [x^1]F(0) \neq 0, F(G(x)) = G(F(x)) = x$，则我们称 $G(x)$ 为 $F(x)$ 的**复合逆**。记为 $G(x) = F^{-1}(x)$。稍后将说明复合逆的存在性和唯一性。

对于两个有限次数的多项式 $F_1(x), F_2(x)$，和非零多项式 $G(x)$，若 $\frac{F_1(x)-F_2(x)}{G(x)}$ 也是一个多项式，则称 $F_1(x) \equiv F_2(x) \pmod{G(x)}$。特别的，若两个多项式模 x^n 同余，则它们的 $x^0, x^1, \cdots, x^{n-1}$ 项系数相同。

在之后的介绍中，我们约定 F 是一类性质比较好的域，使得次数为 n 次的多项式乘法、求逆、求 exp、求 ln 都可以通过 DFT 操作在 $O(n \log n)$ 的复杂度内完成。例如: $F = F_{998244353}$ 时，因为 $998244353 = 7 \cdot 17 \cdot 2^{23} + 1$，$F$ 存在 2^{23} 次单位根，且加减乘运算均可以 $O(1)$ 实现，不存在精度问题，在大部分情况下可以支持除法。

2　多项式牛顿迭代

给定一个次数不大于 n 的多项式 $F(x)$，我们要解出满足 $F(G(x)) = 0$ 的生成函数的前 n 项。

一种常见的方法是使用多项式牛顿迭代去倍增求解。

假设我们已经得到了 $F(A(x)) \equiv 0 \pmod{x^m}$，试图找到 $B(x)$ 使得 $F(B(x)) \equiv 0 \pmod{x^{2m}}$。。那么我们由泰勒展开的公式：

$$F(B(x)) = F(A(x)) + F'(A(x))(B(x) - A(x)) + \frac{F''(A(x))}{2!}(B(x) - A(x))^2 + \cdots \tag{6}$$

考虑在 $\bmod x^{2m}$ 意义下，我们有 $F(B(x)) \equiv F(A(x)) + F'(A(x))(B(x) - A(x)) \pmod{x^{2m}}$。

从而只需令 $B(x) = A(x) - \frac{F(A(x))}{F'(A(x))}$ 即可！

如果我们知道 $G(x)$ 在 $\bmod x$ 下的一个解，那么通过如下迭代，可以得到 $G(x) \bmod x^2$, $\bmod x^4$, $\bmod x^8, \cdots$ 的结果。因此做 $O(\log n)$ 次迭代即可得到 $G(x) \bmod x^{n+1}$ 的结果！

如果 $F(G(x))$ 容易计算，例如 $F(x)$ 是个低次多项式，或者容易用 ln, exp 表示，那么该算法的复杂度往往可以达到 $O(n \log n)$。

在某些情况下，F 不必局限于一个多项式。只要能够计算出 $G(x) \mod x$ 的结果，每次迭代的操作是有意义的，那么 F 可以推广为形如 $F(G(x), x)$ 的二元函数。例如 $F(x) = (x+1)G(x)+1, F(x) = (x+1)G(x)^2 + \left(x^2+2x+2\right)G(x)+1$ 等。而这时对 $F(G(x), x)$ 应该看作是对 $G(x)$ 求偏导！

此处我们先介绍基本的两种牛顿迭代，之后我们将介绍用牛顿迭代法解多项式微分方程的例子。

3　多项式的复合逆

对于 $[x^0]F(x) = 0, [x^1]F(x) \neq 0$ 的生成函数 $F(x)$，称满足 $G(F(x)) = x$ 的函数 G 为它的左逆元，称满足 $F(H(x)) = x$ 的函数 H 为它的右逆元。

设 $G(x) = a_0 + a_1x + a_2x^2 + \cdots$，我们递推构造 $a_0, a_1, a_2, \cdots$。令 $a_0 = 0, a_1 = \frac{1}{[x^1]F(x)}$。比较 $G(F(x))$ 的 x^k 系数 $(k \geq 2)$，那么我们有：

$$[x^k]a_0 + [x^k]a_1F(x) + [x^k]a_2F^2(x) + \cdots + [x^k]a_{k-1}F^{k-1}(x) + a_k\left([x^1]F(x)\right)^k = 0 \tag{7}$$

这是因为 $F^{k+1}(x), F^{k+2}(x)$ 均不含 x^k 项系数，且 $F^k(x)$ 的系数就是 $\left([x^1]F(x)\right)^k$。

因此，我们知道 $F(x)$ 的左逆 **存在且唯一**。

另一方面，设 $H(x) = b_0 + b_1x + b_2x^2 + \cdots, b_0 = 1, b_1 = \frac{1}{[x^1]F(x)}$。同样比较 $F(G(x))$ 的 x^k 系数，我们有：

$$[x^k]F\left(b_0 + b_1x + \cdots + b_{k-1}x^{k-1}\right) + \left([x^1]F(x)\right)^k b_k = 0 \tag{8}$$

因此，$F(x)$ 的右逆 **存在且唯一**。

且注意到 $G(x) = G(F(H(x))) = H(x)$，故 $F(x)$ 的左逆与右逆是相等的，我们统称为 **复合逆**。在此我们证明了 $F(x)$ 的复合逆存在且唯一。

4　拉格朗日反演

拉格朗日反演公式: 若 $F(x)$ 是 $G(x)$ 的复合逆，则

$$[x^n]G(x) = \frac{1}{n}[x^{-1}]\left(\frac{1}{F(x)}\right)^n \tag{9}$$

推广形式为：

$$[x^n]H(G(x)) = \frac{1}{n}[x^{-1}]H'(x)\left(\frac{1}{F(x)}\right)^n \tag{10}$$

其中 $n \geq 1$。

下面给出证明：

设 $H(G(x)) = \sum_{i=0}^{+\infty} a_i x^i$。

那么我们有：

$$\sum_{i=0}^{+\infty} a_i F(x)^i = H(x) \tag{11}$$

对 (11) 两边求导得：

$$\sum_{i=1}^{+\infty} i a_i F(x)^{i-1} = H'(x) \tag{12}$$

(12) 两边除以 $F^n(x)$，那么我们发现，$\frac{1}{F^2(x)}, \frac{1}{F^3(x)}, \cdots$，均不对 x^{-1} 项系数产生贡献，且 $F(x), F^2(x), \cdots$ 也不对 x^{-1} 项系数产生贡献，故提取 x^{-1} 项系数，我们有：

$$n[x^n]H(G(x)) = [x^{-1}]H'(x)F(x) \tag{13}$$

这样即可证明出我们的命题。

5 组合计数问题分析

5.1 用拉格朗日反演求关于复合逆的式子的某一项

在这一部分，我们往往会把问题转换成：已知一个多项式 $F(x)$ 的复合逆 $G(x)$，想要求 $H(G(x))$ 中的某一项的值。我们常用的方法是使用拉格朗日反演，将问题转换为求 $F(x)^n$ 的问题！

例 1 推导含有 n 个点的生成树个数。

解: 这里我们给出一种与传统组合方法不同的方法。

设 $T(x)$ 表示含有 n 个顶点的有标号的 **有根树**个数组成的 EGF。

那么我们删去根后，得到若干个有根树的划分，根据多项式 exp 的组合定义，我们有：$T(x) = xe^{T(x)}$。

令 $F(x) = \frac{x}{e^x}$，从而 $T(x)$ 是 $F(x)$ 的复合逆。

则我们由拉格朗日反演

$$
\begin{aligned}
&[x^n]T(x) && (14)\\
&= [x^{-1}]\frac{1}{F^n(x)} && (15)\\
&= [x^{-1}]\left(\frac{e^{nx}}{x^n}\right) = n^{n-1} && (16)
\end{aligned}
$$

再将 n^{n-1} 除以 n，即可得到 n 个顶点带标号生成树的个数为 n^{n-2}

例 2 给定 $\{1, 2, \cdots, s\}$ 的一个子集 D。对于一棵带有正整数点权的有根多叉树，如果它满足这样的性质，我们就称它为好的：点权为 1 的结点是叶子结点；对于任一点权大于 1 的结点 u，u 的孩子数目 deg[u] 属于集合 D，且 u 的点权等于这些孩子结点的点权之和。给出一个整数 $s\left(1 \le s \le 10^5\right)$，你需要求出根节点权值为 s 的好的多叉树个数，这里一个点的儿子是有序的。我们只需要知道答案关于素数 $950009857 = 453 \cdot 2^{21} + 1$ 取模后的值。

解: 令权值为 n 的好的有根树个数的 OGF 为 $T(x)$，那么我们会发现要么有根树要么恰好有 1 个顶点，要么可以表示为若干棵子树序列。

因此，我们有:

$$T(x) = x + \sum_{d \in D} T(x)^d \tag{17}$$

令 $F(x) = 1 - \sum_{d \in D} x^d$，同样的，我们会发现 $T(x)$ 是 $F(x)$ 的复合逆。

那么 $[x^s]T(x) = [x^-1]\frac{1}{F(x)^s}$。运用一次多项式 ln 和多项式 exp 即可得出原题的答案，其时间复杂度为 $O(s \log s)$。

例 3 求出 n 阶点双、边双连通图的个数，其中顶点有标号，且图不含重边和自环。

解: 设 $C(x)$ 表示 n 阶连通无向图的个数 **乘以 n** 的 EGF。$A(x)$ 为 $n+1$ 阶的点双连通图个数的 EGF, $B(x)$ 为 n 阶边双连通图个数 EGF。这里 $C(x)$ 可以用多项式 ln 的方法在 $O(n \log n)$ 的时间复杂度内求出。

先求 $A(x)$。我们用 $A(x)$ 去表示 $C(x)$。$C(x)$ 的每一项的意义相当于是**含有一个代表顶点的连通图**, 可以类比有根树。

取这个代表顶点 u 并删去,它和它所在的边,设它构成若干个连通分量 $C_1, C_2, C_3, \cdots, C_m$。

其中 $C_i \cup U$ 对应了一个 u 在原图中的点双连通分量 D_i。再删去 D_i 内部的边之后，对应了恰好 D_i 个连通分量。则 C_i 的选法除以 $|C_i|!$ 为: $[x^{|D_i|}]A(x) \cdot [x^{|C_i|}]C(x)^{|D_i|}$。

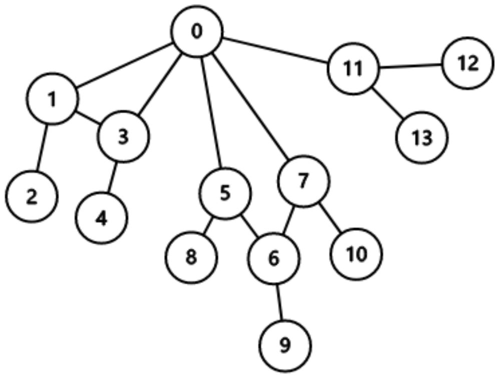

图 1:

如图所示，这里 0 视作代表顶点，三个点双连通分量的大小为 3, 4, 2。且删去一个点双连通分量内部的连边之后，会产生这个该点双连通分量大小 - 1 的不含 0 的连通块。

结合多项式 exp 的组合意义，我们有:

$$C(x) = xe^{A(C(x))} \tag{18}$$

设 $H(x) = \ln \frac{C(x)}{x}$，则我们有：

$$A(x) = H\left(C^{-1}(x)\right) \tag{19}$$

对 (19) 式运用 (10) 式所述的扩展拉格朗日反演即可！

再求 $B(x)$。同样用 $B(x)$ 表示 $C(x)$。我们把 n 阶连通图的代表顶点 u 所在的极大边双连通分量 S 内部的边删去，这样会把图分割为若干个连通块，其中每个连通块与 S 的交集大小为 1。我们如果选取每个连通块的代表顶点为与 S 相交的点，那么将这个代表顶点的任何一条边删去后，图不连通。故这个连通块的取法的 EGF 应该为：$\left(xe^{C(x)}\right)$。

下图展示了一种典型的情形。

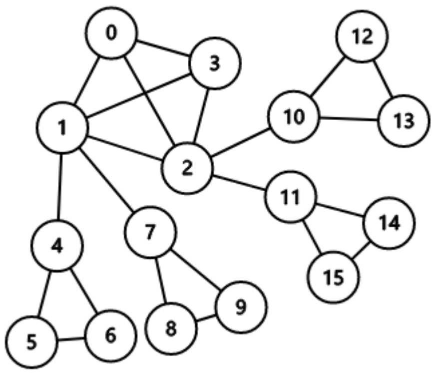

图 2:

与图 1 不同的是，代表顶点 0 恰好在一个边双内，而不能在多个边双。
且由于边 $(1,4),(1,7),(2,11),(2,10)$ 都是桥，故 $4,5,6$ 这三个点与 $7,8,9$ 之间不能连边。
在统计 $1,4,5,6,7,8,9$ 这 7 个点的连接情况时，不能当成连通图的个数来计算，而是看成删去 1 后，代表顶点为 4 的一个连通块和代表顶点为 7 的一个连通块。

故我们按 u 所在的边双连通分量大小分类，可以得到：

$$C(x) = \sum_{i=1}^{+\infty} \left([x^i]A(x)\right)\left(xe^{C(x)}\right)^i \tag{20}$$

也就是说：

$$C(x) = B\left(xe^{C(x)}\right) \tag{21}$$

这里设 $G(x)$ 为 $xe^{C(x)}$ 的复合逆,就只需套用推广形式的拉格朗日反演来计算 $[x^n]C(G(x))$ 即可！

故我们均可以在 $O(n\log n)$ 的时间内计算点数为 n 的点双连通图、边双连通图的个数（最后答案都要除以 n)。

例 4 (LOJ 6363)

解: 给定 $\{1,2,\cdots,n\}$ 的子集 S，我们要求大小为 n 的所有极大点双连通分量都在 S 中的无向连通图的个数。$(n,\sum_{x\in S}x\le 100000)$

我们用例 3 的结果，对任意 $x\in S$，可以求出点数为 x 的点双连通图的个数 e_x。

设 $E(x)=\sum_{i\in S}\frac{e_i}{i!}x^i$。设**含有代表顶点**的极大点双连通分量个数都在 S 中的图的 EGF 为 $F(x)$。

那么用类似例 3 的推导，我们有：

$$F(x)=xe^{E(F(x))} \tag{22}$$

则 $F(x)$ 为 $\frac{x}{e^{E(x)}}$ 的复合逆，使用简单形式的拉格朗日反演，即可重新推出满足条件的连通图的个数！

时间复杂度为 $O((n+\sum_{x\in S}x)\log n)$。

5.2　用复合逆求解关于多项式幂次的式子

在最近出现的很多拉格朗日反演的问题里面，很多都可以抽象地转换为: 对任意 $i=0,1,2,\cdots,n$，求出

$$[x^k]F(x)G(x)^i \tag{23}$$

这里 $F(x),G(x)$ 都是次数为常数或者容易用 exp, ln 等基本函数表示的生成函数，但 G 可能会存在 $x^{-1},x^{-2},\cdots,x^{-n}$ 项，$k=\Theta(n)$。

对于这一类问题，我们考虑引入一个新的变元 u。则答案序列可以表示为：

$$F(x)+uF(x)G(x)+u^2F(x)G(x)^2+\cdots=\frac{F(x)}{1-uG(x)} \tag{24}$$

现在考虑用拉格朗日反演提取 (23) 式中的 x^k 系数。先假设 $G(x)$ 存在复合逆 $H(x)$，$P(x)=F(H(x))$ 那么我们会得到:

$$[x^k]\frac{F(x)}{1-uG(x)}=[x^k]\frac{P(G(x))}{1-uG(x)}=\frac{1}{k}[x^{-1}]\frac{P'(x)(1-ux)+uP(x)}{(1-ux)^2}\left(\frac{1}{H(x)}\right)^k \tag{25}$$

因此，我们只需计算 $G(x)$ 的复合逆，以及 $F(H(x))$，就可以在 $O(n\log n)$ 的时间内求出结果。当 $F(x)=1$ 时，这个问题可以在 $O(n\log n)$ 的代价内与复合逆相互转化。

另外，如果 $G(x)$ 不存在复合逆，则需要对 $G(x)$ 做一些处理。

若 $G(x)$ 的常数项不为 0，设 $G(x) = c + Q(x)$。则我们先求出 $[x^k]F(x)Q(x)^i$，然后注意到：

$$[x^k]F(x)(c+Q(x))^i = [x^k]\sum_{j=0}^{i}\frac{i!}{j!(i-j)!}c^jF(x)Q(x)^{i-j} \tag{26}$$

因此，只需要用 $O(n\log n)$ DFT 求一次卷积即可！

下面再处理 $G(x)$ 的最低次项不为 x^1 的问题。设 $G(x)$ 的最低次项为 g_tx^t。

那么对 $\frac{G(x)}{g_tx^t}$ 开 t 次方，可以将 $G(x)$ 表示为 $q_tQ(x)^t$。

故我们把问题转化为了求 $[x^k]\left(F(x)Q(x)^{it}\right)$。注意到我们只关心 $it \le k$ 的那些项，因此用 $Q(x)$ 代替前述的 $G(x)$ 作讨论，将 n 与 k 取 min，即可做到类似的复杂度。

只要能够方便地求出复合逆和多项式复合，就可以在 $O(n\log n)$ 的时间内解决这个问题。而很多题目里面 $F(x), G(x)$ 都是可以用 exp, ln 以及幂次来表示的函数。

例 5, ZJOI2020 抽卡

当期卡池中共有 m 张不同的卡，每次抽卡，Bob 都可以等概率随机获得一张卡池中的卡。如果 Bob 抽到了一张他已经拥有的卡，那么什么事都不会发生，等于 Bob 浪费了这次抽卡机会。

Bob 是个谨慎的人，他想知道，如果他不停地抽卡直到抽到编号连续的 k 张卡时停止抽卡，期望需要抽多少轮。

输入这 m 张不同的卡的编号 $a_1, a_2, \cdots, a_m \le 2m$。

数据满足 $1 \le m \le 200000, 2 \le k \le m$。输出答案模 998244353 的结果。

解: 首先，对于一个选了 i 张卡牌，且没有 k 张卡牌编号连续的状态，容易计算出它出现的概率和出现它所用的期望轮数。因此我们只需对每个 i 求出含有 i 张卡牌，没有 k 张卡牌连续的编号总数。

事实上，我们只把 $a_1, a_2, \cdots, a_m$ 划分成若干连续段之后，就只需要求每个连续段所对应的答案，再用一次分治多项式乘法将它们乘起来即可。

对一个长度为 n 的连续段 $1, 2, \cdots, n$。我们补充添加两张必定不选的卡牌 $0, n+1$，然后对与每张不选的卡牌 (除了 $n+1$)，记录它后面那张不选的卡牌到它的距离。题目条件等价于这些距离之和为 $n+1$，且每个距离都 $\le k$。

因此，我们只需要对每个 i，求出：

$$[x^{n+1}](\frac{x-x^{k+1}}{1-x})^i \tag{27}$$

这就转为了前述的问题了！这里我们发现 $\frac{x-x^{k+1}}{1-x}$ 是容易用牛顿迭代求出复合逆的一个多项式，因此可以在 $O(m\log m)$ 的时间内算出 (27) 式。而分治多项式乘法需要 $O(m\log^2 m)$ 的

时间，故总复杂度为 $O(m\log^2 m)$。

然而，有些时候所求的式子需要经过变形，才能转换成形如 (23) 式的问题。

例 6, Codeforces Round #641 F2

定义一个长度为 n 的序列 $p_1, p_2, \cdots, p_n$ 是好的，当且仅当 $p_1, \cdots, p_n$ 都是 1 到 n 的整数，且对任意 $k > 1$，存在 $1 \le i < j \le n$ 使得 $p_i = k-1, p_j = k$。

要求对于每个 $k = 1, 2, \cdots, n$，求出所有好的序列 k 出现的次数的总和。

解: 首先，我们把 $p_i = 1$ 的所有 i 从大到小写出来，再把 $p_i = 2$ 的所有 i 从大到小写出来，这样不断地写，写到 $p_i = n$ 的所有 i，会得到一个排列 $q_1, q_2, \cdots, q_n$。

由题目的条件，好的序列由这样的排列唯一确定，这是因为所有 $p_i = j$ 的下标恰好构成 $q_1, q_2, \cdots, q_n$ 的一个连续递减的段。且 p_{q_i} 的值恰好是满足 $p_j < p_{j+1}, 1 \le j < q_i$ 的 j 的个数加一。

设 $d_{i,j}$ 表示在长度为 $i+1$ 的排列中先选择 j 个相邻项要求前一项小于后一项，再填出这个排列的总方案数。把每个连续的小于号当成一段，就可以转换为把 $i+1$ 个数划分成 $i-j+1$ 个集合。容易得到：

$$d_{i,j} = (i+1)![x^{i+1}](e^x - 1)^{i-j+1} \tag{28}$$

求解答案时，我们考虑每个位置的贡献，那么我们有：

$$Ans_{i+1} = \sum_{x=0}^{n-1}\sum_{y=i}^{x}(-1)^{y-i}\binom{y}{i}d_{x,y}\frac{n!}{(x+1)!} \tag{29}$$

$$= \frac{n!}{i!}\sum_{x=0}^{n-1}\sum_{y=i}^{x}(-1)^{y-i}\frac{y!d_{x,y}}{(y-i)!(x+1)!} \tag{30}$$

$$= \frac{n!}{i!}\sum_{y=i}^{n-1}\frac{(-1)^{y-i}y!}{(y-i)!}\sum_{x=y}^{n-1}[z^{x+1}](e^z - 1)^{x-y+1} \tag{31}$$

只需对每个 y 求出 $\sum_{x=y}^{n-1}[z^{x+1})(e^z - 1)^{x-y+1}$，即可用一次卷积求出答案。

设 $F(z) = \frac{e^z-1}{z}$，则我们考虑：

$$\sum_{x=y}^{n-1}[z^{x+1}](e^z-1)^{x-y+1}=\sum_{x=y+1}^{n}[z^x](e^z-1)^{x-y} \tag{32}$$

$$=[z^y]\sum_{x=1}^{n-y}(\frac{e^z-1}{z})^x \tag{33}$$

$$=[z^y]\frac{1-F(z)^{n-y+1}}{1-F(z)} \tag{34}$$

$$=[z^y]\frac{1}{1-F(z)}-[z^{n+1}]\frac{(zF(z))^{n-y+1}}{1-F(z)} \tag{35}$$

注意到 $\frac{1}{1-F(z)}$ 可以用一次多项式求逆求出，而 $[z^{n+1}]\frac{(zF(z))^{n-y+1}}{1-F(z)}$ 就归约到了 (23) 式类型的问题了。

由于 $zF(z)$ 的复合逆 $G(z)$ 和 $\frac{1}{1-F(G(z))}$ 很容易求出，故该问题仍然可以在 $O(n\log n)$ 的时间内解决！

6 多项式牛顿迭代的高级应用

6.1 含有 $G\left(x^2\right),G\left(x^3\right),\cdots$ 的多项式方程

在第二节中，我们介绍了求解关于 $G(x)$ 的方程 $F(G(x),x)=0$ 的方法。但是，在和问题中（特别是同构意义下树的计数），我们得到了多项式方程可能含有 $G\left(x^2\right),G\left(x^3\right)$ 这些项。这个时候，我们假设得到了 $G(x) \mod x^k$ 的结果，那么 $G(x^2),G(x^3),\cdots$ 在 $\mod x^{2k}$ 的结果已经确定了。故在推导迭代公式时，我们只需要把 $G(x^2),G(x^3),\cdots$ 当作一个常数即可！

例 7. 无标号有根树计数

在同构意义下求所有含有 n 个顶点的有根树个数。

设 $T(x)$ 表示含有 n 个顶点的有根树的个数的 OGF。我们试图得到 $T(x)$ 的一个方程。

注意到所有含有 i 个顶点的，总顶点数为 j 的 **有根树可重集合**的个数为：

$$[x^j](\frac{1}{(1-x^i)^{[x^i]T(i)}}) \tag{36}$$

而 n 个顶点的同构意义下有根树的个数就是总共 $n-1$ 个顶点的有根树可重集合的个数。

因此由多项式乘法的组合意义，我们有：

$$T(x)=x\prod_{i=1}^{n}\frac{1}{(1-x^i)^{[x^i]T(x)}} \tag{37}$$

对上式两边求 ln，再结合泰勒展开式 $-\ln(1-x) = x + \frac{x^2}{2} + \frac{x^3}{3} + \cdots$ 可得：

$$\ln\frac{T(x)}{x} = \sum_{i=1}^{+\infty}\frac{T(x^i)}{i} \tag{38}$$

也即：

$$T(x) = x\cdot\exp(\sum_{i=1}^{+\infty}\frac{T(x^i)}{i}) \tag{39}$$

假设得到了 $A(x) \equiv T(x) \pmod{x^m}$ 的值，希望找到 $B(x) \equiv T(x) \pmod{x^{2m}}$。

把上式右边对 $\sum_{i=1}^{+\infty}\frac{A(x^i)}{i}$
这一点泰勒展开，并忽略其二次、三次项，我们有：

$$B(x) \equiv x\cdot\exp(\sum_{i=1}^{+\infty}\frac{A(x^i)}{i})(1+B(x)-A(x)) \pmod{x^{2m}} \tag{40}$$

这是一个关于 $B(x)$ 的线性方程，很容易通过一次多项式求逆解出 $B(x)$。且由调和级数的性质，计算 $\sum_{i=1}^{+\infty}\frac{A(x^i)}{i}$ 的复杂度也是 $O(m\log m)$ 的。

总复杂度为 $O(n\log n)$。

6.2 多项式牛顿迭代解微分方程

这一节我们将讨论如何解**一部分** $G'(x) = F(G(x), x)$ 形式的方程。也就是所已知 $[x^0]G(0)$，求 $G(x)$ 的前 n 项。其中 $F(G(x), x)$ 是关于 $G(x), x$ 的二元生成函数。

6.2.1 递推方法

假设我们得到了 $[x^0]G(x), [x^1]G(x), \cdots, [x^{m-1}]G(x)$，那么

$$[x^m]G(x) = \frac{1}{m}[x^{m-1}]G'(x) = \frac{1}{m}[x^{m-1}](F(G(x), x) \tag{41}$$

值得注意的是，这种方法在 $F(G(x), x)$ 极其特殊的时候，能够取得很好的效果。例如 $F(G(x), x) = (x^5 + x^4 + 4x^3 + 5x^2 + x + 4)G(x)$ 等。这样每次递推的代价就变成了常数级别的了。另外，有些问题中，可以把这样的递推用 cdq 分治 NTT 优化成 $O(n\log^2 n)$ 的复杂度。但是对于一般的问题来说，这样做往往不够高效。

6.2.2 牛顿迭代方法

首先，我们得到 $[x^0]G(x), [x^1]G(x)$。

假设得到了 $G(x) \mod x^m$ 的结果 $A(x)$，试图求出 $G(x) \mod x^{2m-1}$ 的结果 $B(x)$。

把 $F(G(x), x)$ 对 $G(x) = A(x)$ 作泰勒展开，并由 $G(x) \equiv B'(x) \pmod{x^{2m-1}}$，我们有：

$$
\begin{aligned}
B'(x) &\equiv \frac{\partial F}{\partial G}(A(x) - B(x), x) + F(A(x), x) & (42)\\
&\equiv P(x)B(x) + Q(x) \pmod{x^{2m-1}} & (43)
\end{aligned}
$$

其中 $P(x), Q(x)$ 是化简得到的结果。

我们设 $R'(x) = P(x)$，且 $[x^0]R(x) = 0$，那么我们有：

$$
(B(x)e^{-R(x)})' \equiv e^{-R(x)}Q(x) \pmod{x^{2m-1}} \tag{44}
$$

我们用一次多项式积分和多项式求逆，多项式除法即可求出 $B(x)$。

因此，若根据 $A(x)$ 求 $P(x)$，$Q(x)$ 这一步可以做到 $O(m \log m)$，那么总复杂度也能做到 $O(m \log m)$ 了。

7 再探复合与复合逆

7.1 多项式复合

前述的内容主要用于计算**特殊情况下**的多项式复合与复合逆的问题，拉格朗日反演往往只能计算关于复合逆的式子中的一项，而牛顿迭代往往只能快速计算特殊多项式的复合逆。接下来我们简要地介绍一般的多项式复合、复合逆的算法。

给定多项式 $F(x), G(x)$，其次数均不超过 n。我们要在 $O\left(n^2\right)$ 时间内计算出 $F(G(x))$ 的前 n 项系数。这个算法的核心是大步小步思想。

令 $B = \lceil \sqrt{n} \rceil$，我们先求出 $G(x)^0, G(x)^1, \cdots, G(x)^B$，复杂度为 $O\left(n^{1.5} \log n\right)$。再求出 $G(x)^B, G(x)^{2B}, \cdots, G(x)^{B^2}$，复杂度仍然为 $O\left(n^{1.5} \log n\right)$。

接下来，我们发现:

$$
F(G(x)) = \sum_{i=0}^{B-1} G(x)^{iB} \sum_{j=0}^{B-1} [x^{iB+j}] G(x)^j \tag{45}
$$

这样，我们只需要用预处理的结果在 $O\left(n^{1.5}\right)$ 内计算 B 个 $\sum_{j=0}^{B-1}[x^{iB+j}]G(x)^j$，然后用 B 次卷积在 $O\left(n^{1.5} \log n\right)$ 的时间内计算出 $F(G(x))$！

总时间复杂度为 $O\left(n^{1.5}\log n+n^2\right)$。事实上，卷积的那一部分由于常数的原因，可能反而比 $O\left(n^2\right)$ 的那一部分慢。可以通过调整 B 的大小来获得实际上更好的效果。

多项式复合还有一个理论上更好的算法，称为 Brent-Kung 算法，
其时间复杂度为 $O\left((n\log n)^{1.5}\right)$。然而由于常数原因，这个算法在时间上往往不能和常数优秀的 $O(n^2)$ 算法拉开差距。

首先，我们令正整数 $m=\Theta(\frac{\sqrt{n}}{\log n})$。设 $G(x)=G_1(x)+G_2(x)$，这里 $G_1(x)$ 被 x^m 所整除。

将 $F(G(x))$ 在 $G_1(x)$ 这一点展开，我们有：

$$F(G(x))=F(G_2(x))+\frac{F'(G_2(x))}{1!}G_1(x)+\frac{F''(G_2(x))}{2!}G_1(x)^2+\cdots \tag{46}$$

我们发现，上式只有前 $\lceil\frac{n}{m}\rceil$ 项对结果有贡献。
故我们只要计算出 $F(G_2(x)),F'(G_2(x)),F''(G_2(x)),\cdots$，然后做 $O(\frac{n}{m})$ 次长度为 $O(n)$ 的卷积即可计算出 $F(G(x))$。后面这一步复杂度为 $O(\frac{n}{m}n\log n)=O((n\log n)^{1.5})$

先考虑计算 $F(G_2(x))$。首先将 $F(x)$ 的系数分为 m 个连续段，每一段长度为 $\Theta(\frac{n}{m})$。那么我们只需要计算一个 $\le\frac{n}{m}$ 次的多项式 $A(x)$ 复合一次 $\le m$ 次的多项式 $B(x)$ 的结果。

这里我们可以用分治 NTT 的方法计算。每次将 $A(x)$ 分为两个次数几乎相等的多项式 $x^kA_1(x)+A_2(x)$。递归计算 $A_1(B(x)),A_2(B(x)),B(x)^k$，用 $O(\deg A(x)\cdot\log n)$ 的代价计算出 $A(B(x)),B(x)^{\deg A(x)}$。故计算 $A(B(x))$ 可以用 $O(n\log^2 n)$ 的代价算出。计算 $F(G_2(x))$ 也可以用 $O(mn\log^2 n)=O((n\log n)^{1.5})$ 的代价来实现。

若得到了 $F^{(k)}(G_2(x))$，那么用一次求导可以求出 $F^{(k+1)}(G_2(x))G_2'(x)$。再用一次多项式求逆即可算出 $F^{(k+1)}(G_2(x))$ 了（计算两多项式除法时，先不断除以 x）。这样我们又用 $O(\frac{n}{m}\log n)=O((n\log n)^{1.5})$ 的代价从 $F(G_2(x))$ 推到了 $F'(G_2(x)),F''(G_2(x)),\cdots$ 这 $\Theta(\frac{n}{m})$ 个多项式。

综上，我们得到了一个复杂度为 $O((n\log n)^{1.5})$ 的多项式复合算法，这被称为 Brent-Kung 算法。

7.2 多项式复合逆

而对于计算 $F(x)$ 的复合逆的问题，我们把它看成多项式方程 $F(G(x))-x=0$，再使用牛顿迭代进行求解即可！这里我们需要通过 $F'(G(x)),F(G(x))$ 来进行迭代。

如果采用 $O\left((n\log n)^{1.5}\right)$ 的多项式复合算法，那么求复合逆的时间复杂度可以表示为：

$$T(n)=T\left(\frac{n}{2}\right)+O\left((n\log n)^{1.5}\right) \tag{47}$$

由 Master 定理，这个算法的时间复杂度仍然为 $O\left((n\log n)^{1.5}\right)$。

8　总结

本文介绍了多项式拉格朗日反演、牛顿迭代的基本理论，通过一定程度上的抽象，给出了比较通用的解法。

计数组合的理论博大精深，本文仅起到抛砖引玉的作用。特别遗憾的是，本文提出的算法有一定的局限性，例如不能计算模小素数、模合数的结果，不能解形式更复杂的微分方程等等。

希望本文能激发选手们对计数组合研究的热情，也希望这方面的知识能够更好、更全面地引入 OI 界。

9　致谢

感谢中国计算机学会提供学习和交流的平台。

感谢教练金靖老师、吴申广老师对我的指导。

感谢父母对我的培育和教诲。

感谢所有帮助我的同学、老师。

参考文献

[1] 金策，《生成函数的运算与组合计数问题》

[2] https://www.luogu.com.cn/blog/yurzhang/solution-p5373, Brent-Kung 算法介绍

[3] http://codeforces.com/blog/entry/77284, 例 5 的英文题解

[4] https://www.luogu.com.cn/blog/Caro23333/codeforces-round-641-div1f-slime-and-sequences-zhong-wen-ti-xie，例 5 的中文题解

[5] https://blog.csdn.net/sslz_fsy/article/details/104846902,

[6] 刘汝佳,《算法竞赛入门经典》, 清华大学出版社。