母函数的性质及应用

2009 国家集训队论文

毛杰明

南京外国语学校

Email:maojm517@163.com

邮政编码:210008

目录

摘要

母函数理论是离散数学的重要方法，是连接离散数学与连续数学的桥梁。它是处理数列以及组合计数问题的有力工具，它的威力在于能用一种统一的方式解决各种不同类型的问题，所以在解决信息学竞赛的一些问题时母函数有着令人惊讶的效果，在优化算法以及证明命题时也有着重要的作用。本文首先介绍母函数的性质，然后结合一些典型的题目展示母函数在信息学竞赛中的应用，最后本文给出了对母函数的一个总结。

关键字

母函数 递推 排列组合

§1. 母函数的性质

§1.1. 定义

母函数是用于对应一个无穷序列的幂级数，一般来说母函数有形式：

$$G(x) = g_0 + g_1 x + g_2 x^2 + \cdots = \sum_{n\geq 0} g_n x^n$$

我们称$G(x)$是序列$< g_0, g_1, g_2, \cdots >$的母函数,下文表示为：

$$G(x) \leftrightarrow < g_0, g_1, g_2, \cdots >$$

举一个例子：

序列$<1,1,1,1,1\cdots>$对应的母函数是$G(x)=1+x+x^2+x^3+\cdots=\sum_{n\geq 0}x^n$

用等比数列求和公式我们知道$G(x)=1+x+x^2+x^3+\cdots=\frac{1}{1-x}$,这里我们称$\frac{1}{1-x}$是这个母函数的闭形式。

需要注意的是，由于收敛问题，这个公式在$|x|\geq 1$时并不成立。但因为我们在考虑母函数的问题时不需要代入 x 的值求出函数值，所以我们在做母函数时并不需要考虑收敛性，本文将忽略母函数的收敛问题。

§1.2. 基本操作

1．放缩：

$$cG(x)=cg_0+cg_1x+cg_2x+\cdots\leftrightarrow<cg_0,cg_1,cg_2,\cdots>$$

2．加减法：

$$F(x)\pm G(x)=(f_0\pm g_0)+(f_1\pm g_1)x+(f_2\pm g_2)x^2+\cdots\leftrightarrow<f_0\pm g_0,f_1\pm g_1,f_2\pm g_2,\cdots>$$

3．右移:

这个操作指将序列向右平移 k 位，并在前 k 位补 0

$$x^kG(x)=g_0x^k+g_1x^{k+1}+g_2x^{k+2}+\cdots\leftrightarrow<\underbrace{0,\cdots,0}_{k个0},g_0,g_1,g_2,\cdots>$$

4.求导：

$$G'(x)=g_1+2g_2x+3g_3x^2+\cdots\leftrightarrow<g_1,2g_2,3g_3,\cdots>$$

求导操作有两个效果，将序列左移一位并使每项乘以它的下标。

举个例子：

设 $G(x)=1+x+x^2+x^3+\cdots=\dfrac{1}{1-x}$

则 $\dfrac{1}{(1-x)^2}=G'(x)=1+2x+3x^2+4x^3+\cdots$

那么闭形式 $\dfrac{1}{(1-x)^2}$ 对应的序列就是 $<1,2,3,4,5,\cdots>$

5．卷积规则：

$$H(x)=G(x)\cdot F(x)=(g_0+g_1x+g_2x^2+\cdots)(f_0+f_1x+f_2x^2+\cdots)$$

则 $H(x)$ 中 x^n 的系数 $h_n=g_0f_n+g_1f_{n-1}+g_2f_{n-2}+\cdots+g_nf_0$

由此也可以证明上面闭形式 $\dfrac{1}{(1-x)^2}$ 对应的序列就是 $<1,2,3,4,5,\cdots>$

卷积规则比较重要的一点是在处理组合问题时，当 $G(x)$ 对应在 A 中选择元素的母函数， $F(x)$ 对应在 B 中选择元素的母函数，则 $G(x)\cdot F(x)$ 对应在 $A\cup B$ 选择元素的母函数。

§1.3. 简单的序列所对应的母函数

由上面的基本操作我们已经可以求出很多序列对应的母函数了。

上文中已经求出了闭形式 $\dfrac{1}{1-x}$ 和 $\dfrac{1}{(1-x)^2}$ 对应序列，我们考虑求一个常用的母函数 $\dfrac{1}{(1-x)^m}$ 对应的序列：

$$G(x)=\frac{1}{(1-x)^m}=\underbrace{\frac{1}{1-x}\times\cdots\frac{1}{1-x}}_{m个}=(1+x+x^2+\cdots)^m$$

那么 $G(x)$ 中 x^n 的系数 g_n 的值就等价于不定方程 $x_1+x_2+\cdots+x_m=n$ 的非负整数解

的个数，而这个不定方程非负整数解的个数是我们熟悉的问题，由插板法或者其他方法易得 $g_n = C_{m+n-1}^{m-1}$。

我们也可以用另外一种方法求 g_n，$G(x) = \frac{1}{(1-x)^m} = \frac{1}{1-x} \times \frac{1}{(1-x)^{m-1}}$，之后运用数学归纳法同样可以得到结果 $g_n = C_{m+n-1}^{m-1}$。

那么闭形式 $\frac{1}{(1-x)^m}$ 对应的序列为 $<1, C_m^{m-1}, C_{m+1}^{m-1}, C_{m+2}^{m-1}, \cdots>$ 。

我们再来考虑一下闭形式 $\frac{1}{1-\alpha x}$，我们可以把 αx 看成一个整体后来展开，参考 $\frac{1}{1-x}$ 的展开后我们得到 $\frac{1}{1-\alpha x} = 1 + \alpha x + \alpha^2 x^2 + \alpha^3 x^3 + \cdots \leftrightarrow <1, \alpha, \alpha^2, \cdots>$。

结合 $\frac{1}{(1-x)^m}$ 我们可以得到一个比较有用的结论：

$$\frac{1}{(1-\alpha x)^m} = 1 + \alpha C_m^{m-1} x + \alpha^2 C_{m+1}^{m-1} x^2 + \alpha^3 C_{m+2}^{m-1} x^3 + \cdots$$

§1.4. 指数型母函数

有时候序列 $<g_n>$ 所具有的母函数的性质十分复杂，而序列 $<\frac{g_n}{n!}>$ 所具有的母函数的性质十分简单，那我们宁愿选择 $<\frac{g_n}{n!}>$ 来研究，然后再乘以 $n!$。

我们称：

$G(x) = \sum_{n \geq 0} g_n \frac{x^n}{n!}$ 为序列 $<g_0, g_1, g_2, \cdots>$ 的指数型母函数。

因为指数型母函数在处理排列问题时符合上文提到的卷积规则，所以我们一般用指数型母函数来处理排列问题。

我们先来研究一下最基本的序列$<1,1,1,1,\cdots>$所对应的指数型母函数，

它对应的指数型母函数根据定义为：

$$G(x)=1+\frac{x}{1!}+\frac{x^2}{2!}+\frac{x^3}{3!}+\frac{x^4}{4!}+\cdots$$

$G(x)$的闭形式用初等方法不容易得到，这里我们用微积分中的 Taylor 公式：

（不考虑收敛问题时）

$$f(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\frac{f'''(x_0)}{3!}(x-x_0)^3+\cdots$$

取 $f(x)=e^x, x_0=0$，得$e^x=1+x+\frac{x^2}{2!}+\frac{x^3}{3!}+\frac{x^4}{4!}+\cdots=G(x)$，

也就是说序列$<1,1,1,1,\cdots>$的指数型母函数的闭形式为e^x。

同样运用 Taylor 公式，我们可以得到：

序列$<1,-1,1,-1,1,-1,\cdots>$的指数型母函数为e^{-x}。

序列$<0,1,0,1,0,1,\cdots>$的指数型母函数为$\frac{e^x-e^{-x}}{2}$。

序列$<1,0,1,0,1,0,\cdots>$的指数型母函数为$\frac{e^x+e^{-x}}{2}$。

指数型母函数的放缩与加减法操作与普通母函数是一致的，但左移和右移操作的方式不太一样：

$$G(x)=g_0+\frac{g_1}{1!}x+\frac{g_2}{2!}x^2+\cdots$$

则$G'(x)=g_1+\frac{g_2}{1!}x+\frac{g_3}{2!}x^2+\cdots$

也就是说求导可以完成对指数型母函数的左移操作。

同样的道理，积分操作可以完成对指数型母函数的右移操作。

指数型母函数和普通母函数一样，最有趣的运算是乘积。

我们设$G(x)=g_0+\frac{g_1}{1!}x+\frac{g_2}{2!}x^2+\cdots$是序列$<g_{0,}g_1,g_2,\cdots>$的指数型母函数，

$F(x)=f_0+\frac{f_1}{1!}x+\frac{f_2}{2!}x^2+\cdots$是序列$<f_0,f_1,f_2,\cdots>$的指数型母函数。

那么，

$$H(x)=G(x)\cdot F(x)=(g_0+\frac{g_1}{1!}x+\frac{g_2}{2!}x^2+\cdots)(f_0+\frac{f_1}{1!}x+\frac{f_2}{2!}x^2+\cdots)$$

$$=\sum_{n\geq 0}\frac{x^n}{n!}\sum_{k\geq 0}(\frac{g_k}{k!}\times\frac{f_{n-k}}{(n-k)!}\times n!)=\sum_{n\geq 0}\frac{x^n}{n!}\sum_{k\geq 0}(g_k\times f_{n-k}\times C_n^k)$$

其中$H(x)$的x^n的系数$h_n=\sum_{k\geq 0}(g_k\times f_{n-k}\times C_n^k)$。

而我们发现考虑从两种元素中分别取 k 个和 n-k 个时，如果不忽略两种元素的排列位置，有C_n^k种方法，联系上面h_n的值后我们就知道为什么指数型母函数适用于排列问题了。

§1.5. 母函数型 Pólya 定理

Pólya 定理现在已经成为大家熟知的定理了，2001 年符文杰和 2008 年陈瑜希的论文都介绍的是 Pólya 定理，它的具体描述如下：

（Pólya 定理）设$\overline{G}$是 n 个对象的一个置换群，用 m 种颜色涂染这 n 个对象，则不同染色方案数为$l=\frac{1}{|G|}[m^{c(a_1)}+m^{c(a_2)}+\cdots+m^{c(a_g)}]$

我们发现这个定理只能用于计数，对于具体状态的情况并不了解。

例如我们用b,w两种颜色来染 4 个球，那么一共有 16 种情况，可写为：

bbbb, bbbw, bbwb, bbww, bwbb, bwbw, bwwb, bwww,
wbbb, wbbw, wbwb, wbww, wwbb, wwbw, wwwb, wwww

如果我们不关心具体哪个球染哪个颜色时，比如 $wbbw$ 就可以缩写为 b^2w^2 。

这样的话整个方案可以写成 $(b+w)^4$ 的形式，由于 4 个球没有区别，乘法是可以交换的。

$$(b+w)^4 = b^4 + 4b^3w + 6b^2w^2 + 4bw^3 + w^4$$

可用这种方法来推广 Pólya 定理，称为母函数型 Pólya 定理。

我们将 $l = \frac{1}{|G|}[m^{c(a_1)} + m^{c(a_2)} + \cdots + m^{c(a_g)}]$ 中的 $m^{c(a_i)}$ 用

$$(b_1 + b_2 + \cdots + b_m)^{c_1(a_1)}(b_1^2 + b_2^2 + \cdots + b_m^2)^{c_2(a_i)} \cdots (b_1^n + b_2^n + \cdots + b_m^n)^{c_n(a_i)}$$

代换得

$$P = \frac{1}{|G|}\sum_{i=1}^{g}(b_1 + b_2 + \cdots + b_m)^{c_1(a_1)}(b_1^2 + b_2^2 + \cdots + b_m^2)^{c_2(a_i)} \cdots (b_1^n + b_2^n + \cdots + b_m^n)^{c_n(a_i)}$$

其中 $c_j(a_i)$ 表示置换 a_i 中长度为 j 的循环节的个数。

举一个例子：

有 4 颗珠子 v_1, v_2, v_3, v_4 绕成一圈，用 b, w 两种颜色染色，旋转后重叠的方案算一种。

则置换群 G 为

$$(v_1)(v_2)(v_3)(v_4), (v_2\ v_3\ v_4\ v_1), (v_1\ v_3)(v_2\ v_4), (v_4\ v_1\ v_2\ v_3)$$

则利用 Pólya 定理,得不同的方案数为：

$$l = \frac{1}{4}(2^4 + 2 + 2^2 + 2) = 6$$

用母函数型 Pólya 定理得具体的方案为：

$$P = \frac{1}{4}[(b+w)^4 + (b^4 + w^4) + (b^2 + w^2)^2 + (b^4 + w^4)]$$
$$= b^4 + b^3w + 2b^2w^2 + bw^3 + w^4$$

即 2 黑 2 白的有两种，4 黑，4 白，3 黑 1 白，3 白 1 黑的都只有 1 种。

§2. 母函数的应用

§2.1. 原创题

例 1.小明出门旅游，需要带一些食物，包括薯片，巧克力，矿泉水，汉堡，牛奶和糖果。经过估计，他觉得带 $n(n \le 10^{100})$ 件食物比较合适，但他还有一些癖好：

最多带 1 个汉堡

巧克力的块数是 5 的倍数

最多带 4 瓶矿泉水

薯片的包数是一个偶数

最多带 3 罐牛奶

糖果的个数是 4 的倍数

问你小明有多少种方式来准备这次旅行所带的食物。

[分析]

处理本题一般的思路是首先考虑 1，3，5 三个条件，可以用枚举来解决，之后问题转化成了一个不定方程非负整数解的问题：$5x+2y+4z=m$

首先想到的方法是用 $f(m)$ 表示 $5x+2y+4z=m$ 非负整数解的个数,把 3 个变量分开来考虑做 3 次时间复杂为 $O(n)$ 的递推。

还有一种想法是枚举 x 的取值，然后对剩下的方程先用欧基里德扩展算法求出一组解，再算出解的个数。

以上两种算法的时间复杂度无法满足题目中 n 的大小，而且第 2 种方法在条件数增多后

没有推广性。

我们再来看看母函数是怎样解决本题的：

汉堡可以不带或者带一个，对应的母函数：

$$h(x)=1+x$$

巧克力必须 5 的整倍数个带，可以带 0 个，5 个，10 个……，巧克力对应的生成函数为：

$$c(x)=1+x^5+x^{10}+x^{15}+\cdots=\frac{1}{1-x^5}$$

薯片也要 2 的整倍数包带，也可以写出它的生成函数：

$$p(x)=1+x^2+x^4+x^6+\cdots=\frac{1}{1-x^2}$$

矿泉水至少带 4 瓶，生成函数为：

$$w(x)=1+x+x^2+x^3+x^4=\frac{1-x^5}{1-x}$$

最多带 3 罐牛奶，生成函数为：

$$m(x)=1+x+x^2+x^3=\frac{1-x^4}{1-x}$$

糖果的个数是 4 的倍数，生成函数为：

$$s(x)=1+x^4+x^8+x^{12}\cdots=\frac{1}{1-x^4}$$

运用卷积规则，得到小明带 n 件食物选择方案的生成函数：

$$\begin{aligned}&h(x)c(x)p(x)w(x)m(x)s(x)\\&=(1+x)\frac{1}{1-x^5}\frac{1}{1-x^2}\frac{1-x^5}{1-x}\frac{1-x^4}{1-x}\frac{1}{1-x^4}\\&=\frac{1}{(1-x)^3}\\&=1+C_3^2x+C_4^2x^2+C_5^2x^3+\cdots\end{aligned}$$

发现最后竟然就只剩下一个很简单的形式，前面的预备知识中已经解决了它， x^n 的系数就是C_{n+2}^2,也就是本题要求的答案。

这样我们运用母函数的方法解决了本问题，时间复杂度仅为 $O(1)$ 而且基本没有编程难度，在 n 比较大的时候需要使用高精度乘法。

本题的答案比较特殊，可能一些同学会觉得找规律的方法会更加的快捷，但对于一些结果并不是很简洁的题目时，这种找规律的方法就基本无效了。

［小结］

本问题充分体现了母函数的优越性，限制条件被转化成了我们容易处理的东西，并用十分机械化的手法解决了本问题。通过本题，我们发现运用母函数不仅能降低程序的时间复杂度和代码量，还能减少思维难度。

§2.2. Chocolate

例 2. ACM/ICPC Regional Contest Beijing 2002 Problem F Chocolate

一个口袋中装有巧克力，巧克力的颜色有 c 种。现从口袋中取出一个巧克力，若取出的巧克力与桌上已有巧克力颜色相同，则将两个巧克力都取走，否则将取出的巧克力放在桌上。设从口袋中取出每种颜色的巧克力的概率均等。求取出 n 个巧克力后桌面上剩余 m 个巧克力的概率。 $c \le 100, n, m \le 1000000$ 。

[分析]

首先考虑只有当 $2 \mid (m-n)$,且 $m \le n, m \le c$ 时本问题的解不为 0。下面只考虑解不为 0 的情况。

本题容易想到的方法是递推,用 $f[i,j]$ 表示取出 i 块巧克力后，桌面上剩余 j 块巧克力的概率，然后通过 $f[i,j]=\frac{c-j+1}{c}f[i-1,j-1]+\frac{j+1}{c}f[i-1,j+1]$ 进行递推求解。

这种方法的时间复杂度为 $O(nc)$，在题目中所给的数据范围下很容易超时。

我们再来考虑考虑用母函数来解决本问题：

我们来求概率的方法是把所有状况分成若干个等概率的状况，看所求的占了多少种状况。由于题目中说每次从口袋中取出每种颜色的巧克力概率相等，所以总状况按照 n 次取的巧克力分为 c^n 种状况，这种分法是考虑到不同颜色巧克力之间的排列关系的，所以本题适于用指数型母函数来解决。

剩余 m 个巧克力这个条件我们可以转化为 c 种巧克力中有 m 种巧克力取出了奇数次，c-m 种巧克力取出了偶数次。

我们把巧克力分成一种一种考虑，那么那些取出了奇数次的巧克力对应的指数型母函数应该是奇数幂次的系数为 1，偶数幂次的系数幂次为 0，也就是上文提到的序列 $<0,1,0,1,0,1,\cdots>$ 的指数型母函数 $\frac{e^x-e^{-x}}{2}$，同样的道理，那些取出了偶数次的巧克力对应的指数型母函数为 $\frac{e^x+e^{-x}}{2}$。

因此对应 m 种巧克力取出了奇数次，c-m 种巧克力取出了偶数次的指数型母函数为 $G(x)=(\frac{e^x-e^{-x}}{2})^m\times(\frac{e^x+e^{-x}}{2})^{c-m}$，我们要求的状况数即为 $G(x)$ 中 x^n 的系数 $g_n\times n!\times C_c^m$，其中 $n!$ 是指数型母函数需要乘的，C_c^m 是因为 c 种巧克力中不确定是哪 m 个取了奇数次。那么所求的概率为 $\frac{g_n\times n!\times C_c^m}{c^n}$。

下面我们需要把 $G(x)=(\frac{e^x-e^{-x}}{2})^m\times(\frac{e^x+e^{-x}}{2})^{c-m}$ 展开，我们可以把 e^x 看作一个整

体，将$G(x)$展开成关于e^x的多项式的形式，我们考虑展开后每一项e^{kx}（不妨设$k>0$，$k<0$同理）中x^n的系数为$\frac{k^n}{n!}$，把它们加起来即可。

这样，我们可以先把$G(x)=(\frac{e^x-e^{-x}}{2})^m\times(\frac{e^x+e^{-x}}{2})^{c-m}$用$O(c^2)$的复杂度的多项式乘法展开成关于$e^x$的多项式，设$e^{kx}$的系数为$p_k$，那么所求概率：

$$\frac{g_n\times n!\times C_c^m}{c^n}=C_c^m\times\frac{n!}{c^n}\times\sum_{k=1}^{c}\frac{k^n}{n!}\times(p_k+(-1)^n p_{-k})=C_c^m\times\sum_{k=1}^{c}(\frac{k}{c})^n\times(p_k+(-1)^n p_{-k})$$

由于题目要求的精度不是很高，多项式展开时系数可以用实数来算，$(\frac{k}{c})^n$可以用对数来算，并且在n很大$(n\geq 2000)$的时候若$k\neq c$那么$(\frac{k}{c})^n$可以约等于0。

这样本题用母函数的方法解决了，时间复杂度为$O(c^2)$。

［小结］

本题是一道用指数型母函数解决排列问题的典型题目。当用递推方法无法很好的解决本题时，我们用指数型母函数很好的解决了这个问题，再一次体现了母函数能优化算法的功能。本题中的母函数方法是介于对问题一个小小的转化才可以用的，这种转化问题的能力也是我们需要注意的。

§2.3. Sweet

例 3. CEOI2004 Sweet

John 得到了 n 罐糖果。不同的糖果罐，糖果的种类不同（即同一个糖果罐里的糖果种类是相同的，不同的糖果罐里的糖果的种类是不同的）。第 i 个糖果罐里有 m_i 个糖果。John

决定吃掉一些糖果，他想吃掉至少 a 个糖果，但不超过 b 个。问题是 John 无法确定吃多少个糖果和每种糖果各吃几个。有多少种方法可以做这件事呢？

你的任务是写一个程序：

从标准输入读入每罐糖果的数量，整数 a 到 b

确定 John 能够选择的吃掉糖果的方法数（满足以上条件）

把结果输出到标准输出（把答案模 2004 输出）

$(1 \le n \le 10, 0 \le a \le b \le 10000000\ , 0 \le m_i \le 1000000\ )$

[分析]

本题要求的是吃的糖果在 a 到 b 之间的方法数，不是很容易处理。我们转化一下，记 $f(m)$ 为最多吃 m 个糖果的方法数，那我们所求的就是 $f(b)-f(a-1)$。这样我们将问题转化为求 $f(m)$。

本题是一个组合问题。糖果 i 可以吃 0 到 m_i 个,那么糖果 i 的母函数写出来：

$G_i(x)=1+x+x^2+\cdots+x^{m_i}=\dfrac{1-x^{m_1+1_i}}{1-x}$, 那么整个问题的母函数

$$G(x)=G_1(x)\times G_2(x)\times\cdots\times G_n(x)=\frac{(1-x^{m_1+1})\times(1-x^{m_2+1})\times\cdots\times(1-x^{m_n+1})}{(1-x)^n}$$

$$=(1-x^{m_1+1})\times(1-x^{m_2+1})\times\cdots\times(1-x^{m_n+1})\times(1+C_n^{n-1}x+C_{n+1}^{n-1}x^2+C_{n+2}^{n-1}x^3+\cdots)$$

下面我们需要先将 $(1-x^{m_1+1})\times(1-x^{m_2+1})\times\cdots\times(1-x^{m_n+1})$ 展开，由于每一个括号内只有 2 个元素，所以展开这 n 个括号的内容只需要 $O(2^n)$ 的时间复杂度。展开后我们得到一个最多有 2^n 项系数非 0 的多项式 $\sum_{k>0}p_kx^k$ ， p_k 为次数为 k 的项的系数。

由于我们所需要求的 $f(m)$ 定义为最多吃 m 个糖果的方法数,也就是可能吃 $0-m$ 个糖果，所以 $f(m)$ 的值应该是 $G(x)$ 的 $0-m$ 次项系数和，也就是说

$$f(m)=\sum_{k\geq 0}(p_k\times(1+C_n^{n-1}+C_{n+1}^{n-1}+\cdots+C_{n-1+m-k}^{n-1}))$$

下面需要解决的问题就是$1+C_n^{n-1}+C_{n+1}^{n-1}+\cdots+C_{n-1+m-k}^{n-1}$这么一串数如何快速求出。我们熟知一个组合恒等式$C_m^n=C_{m-1}^{n-1}+C_{m-1}^n$，这里可以应用它简化那一串数：

$$1+C_n^{n-1}+C_{n+1}^{n-1}+\cdots+C_{n-1+m-k}^{n-1}=C_n^n+C_n^{n-1}+C_{n+1}^{n-1}+\cdots+C_{n-1+m-k}^{n-1}$$
$$=C_{n+1}^n+C_{n+1}^{n-1}+\cdots+C_{n-1+m-k}^{n-1}=C_{n+2}^n+C_{n+2}^{n-1}+\cdots+C_{n-1+m-k}^{n-1}$$
$$=\cdots$$
$$=C_{n+m-k}^n$$

那么，$f(m)=\sum_{k\geq 0}p_k\times C_{n+m-k}^n$

最后只剩下一个问题了，就是计算C_{n+m-k}^n。由于 n 比较小，我们可以直接把C_{n+m-k}^n的展开成$\frac{(n+m-k)\times(n+m-k-1)\times\cdots\times(m-k+1)}{n\times(n-1)\times\cdots\times 1}$的形式，然后让分子一边乘一边模$2004\times n!$，最后再将这个值除以$n!$即可。这一步只需要$O(n)$的时间复杂度。

这样，本题就用母函数的方法解决了，时间复杂度为$O(2^n\times n)$。

当然，本题也可以用别的方法解决，比如说利用容斥原理。你会发现用容斥原理会得到跟母函数一样的结论。

［小结］

本题是一个组合计数问题，我们用母函数的方法很好的解决了本问题，我们发现母函数在组合问题中的用途十分广泛。本题也可以用容斥原理解决，并且解法的后半部分十分相似，可谓是殊途同归。但是母函数的方法更加一般化，思维难度也比较小。我们虽然很喜欢一些特殊的漂亮的解答，但我们更喜欢更加一般化的方法，因为它能解决更多的问题。

§2.4. 证明题

例 4.用母函数证明 n 个顶点的完全图中有n^{n-2}棵生成树。

[分析]

我们设 n 个顶点的完全图中有t_n棵生成树。

我们来考虑一般情况，即 n 个顶点。如果直接去考虑这 n 个顶点两两间的关系，问题会变得相当复杂。

我们采用常用的手段，考虑其中一个顶点，设这个点为 p。把这个顶点去掉后，生成树被分成了 m 个树，若每个树有k_i个节点，那么它们自身的组合方式以及连接 p 的方式一共有$(k_1 \times t_{k_1}) \times (k_2 \times t_{k_2}) \times \cdots \times (k_m \times t_{k_m})$种，结合整体来考虑，我们可以得到一个关于$t_n$的递推式：

$$t_n = \sum_{m>0} \frac{1}{m!} \sum_{k_1+k_2+\cdots+k_m=n-1} \binom{n-1}{k_1,k_2,\cdots,k_m} \times (k_1 \times t_{k_1}) \times (k_2 \times t_{k_2}) \times \cdots \times (k_m \times t_{k_m})$$

其中$\binom{n-1}{k_1,k_2,\cdots,k_m}$表示把 n-1 个元素分成大小为$k_1, k_2, \cdots, k_m$的 m 个部分的方法数，而除以$m!$是因为我们不考虑部分之间的排序关系。

我们稍微简化以下这个式子后得到：

$$\frac{t_n \times n}{n!} = \sum_{m>0} \frac{1}{m!} \sum_{k_1+k_2+\cdots+k_m=n-1} \frac{t_{k_1} \times k_1}{k_1!} \times \cdots \times \frac{t_{k_m} \times k_m}{k_m!}$$

我们换一下元，设$g_n = t_n \times n$,则可得：

$$\frac{g_n}{n!} = \sum_{m>0} \frac{1}{m!} \sum_{k_1+k_2+\cdots+k_m=n-1} \frac{g_{k_1}}{k_1!} \times \cdots \times \frac{g_{k_m}}{k_m!}$$

设$G(x)$为序列$< g_n >$的指数型母函数，我们有：

$$G(x)=\frac{g_1}{1!}x+\frac{g_2}{2!}x^2+\frac{g_3}{3!}x^3+\cdots=x\sum_{n>0}\frac{1}{m!}\sum_{k_1+k_2+\cdots+k_m=n-1}(\frac{g_{k_1}}{k_1!}\times x^{k_1})\times\cdots\times(\frac{g_{k_m}}{k_m!}\times x^{k_m})$$

$$=x\sum_{m>0}\frac{1}{m!}\sum_{n>0}\sum_{k_1+k_2+\cdots+k_m=n-1}(\frac{g_{k_1}}{k_1!}\times x^{k_1})\times\cdots\times(\frac{g_{k_m}}{k_m!}\times x^{k_m})$$

$$=x\sum_{m>0}\frac{1}{m!}G(x)^m=xe^{G(x)}$$

上式的最后一步运用了上文提到的$e^x=1+x+\frac{x^2}{2!}+\frac{x^3}{3!}+\frac{x^4}{4!}+\cdots$。

这样我们得到了$G(x)=xe^{G(x)}$，我们发现这和广义指数级数$\varepsilon_1(x)$很像。

由$\varepsilon_1(x)\times x=\ln\varepsilon_1(x)$，我们知$G(x)=\varepsilon_1(x)\times x$

我们知道$\varepsilon_1(x)=\sum_{n\geq 0}\frac{(n+1)^{n-1}}{n!}x^n$

这样我们得到$t_n=\frac{g_n}{n}=\frac{n!}{n}\times\frac{n^{n-2}}{(n-1)!}=n^{n-2}$,而上面递推时没有考虑$n=1$,所以要特别判断一下。

命题得证。

[小结]

本题体现了母函数在证明有关信息学竞赛的一些结论时的作用，母函数经常被用来证明一些组合结论。

§2.5. Polygon

例 5.IOI2003 国家集训队难题讨论活动 0015 Polygon

给出 n,m(n≥m)，从单位正 n 边形的顶点中选出 m 个，能组成多少种不同的 m 边形？

（若一个 m 边形可由另一个 m 边形通过旋转、翻转、平移得到，则认为两个 m 边形同种）

[分析]

首先对于这种要求旋转，翻转本质不同的方案数的题目，我们第一个反应就是使用 Pólya 定理。但这个平移的条件对我们构造置换群产生了一大难题，我们发现可以证明不存在正 n 边形的顶点构成的两个 m 边形平移后重合。证明如下：

考虑正 n 边形的 n 个顶点共圆，如果有两个 m 边形平移后重合，那么那个圆上就有 m 条等长且平行的弦。如果用一组平行直线截一个圆，最多只能截出两个等长的弦，而显然 m ≥3,产生矛盾，所以不存在正 n 边形的顶点构成的两个 m 边形平移后重合。

我们只需要考虑旋转，翻转两个条件。

于是问题转化为将正 n 边形的 n 个顶点用b,w染色，b 染了 m 次，求旋转和翻转后本质不同的方案个数。我们发现这个问题直接用 Pólya 定理行不通，需要使用上文提到的母函数型 Pólya 定理。

我们先来求置换群G：（我们对每个置换a,只要求出它的$c_1(a),\cdots,c_n(a)$，因为只有这个在用母函数型 Pólya 定理有用。）

1. 旋转：

旋转的情况比较简单，旋转 i 次时得到的置换$a_i(i=0,1,2,\cdots,n-1)$,只有$c_{\frac{n}{(n,i)}}(a_i)=(n,i)$不为 0。（定义$(n,0)=n$）

2. 翻转：

翻转时有点麻烦，需要分 n 的奇偶讨论。

（1）n 为奇数，以每个顶点以及多边形中心的连线为轴。设以第 i 个点和正 n 边形的中心的连线为轴得到的置换为$b_i(i=1,2,3,\cdots,n)$,那么有$c_1(b_i)=1,c_2(b_i)=\frac{n-1}{2}$，其它的为

0。

（2）n 为偶数，翻转的对称轴有两种：

①轴过多边形的两个顶点，设置换为$b_i(i=1,2,3,\cdots,\frac{n}{2})$,有$c_1(b_i)=2,c_2(b_i)=\frac{n}{2}$-1，其它的为 0。

②轴是多边形两对边中点的连线时，设置换为$b_i(i=\frac{n}{2}+1,\frac{n}{2}+2,\frac{n}{2}+3,\cdots,n)$，有$c_2(b_i)=\frac{n}{2}$，其它的为 0。

下面套用母函数型 Pólya 定理可以得到：

1.当 n 为奇数时：

$$P=\frac{1}{2n}[\sum_{i=0}^{n-1}(b^{\overline{(n,i)}}+w^{\overline{(n,i)}})^{(n,i)}+n(b+w)(b^2+w^2)^{\overline{\frac{n-1}{2}}}]$$

2.当 n 为偶数时：

$$P=\frac{1}{2n}[\sum_{i=0}^{n-1}(b^{\overline{(n,i)}}+w^{\overline{(n,i)}})^{(n,i)}+\frac{n}{2}(b+w)^2(b^2+w^2)^{\overline{\frac{n}{2}-1}}+\frac{n}{2}(b^2+w^2)^{\overline{\frac{n}{2}}}]$$

我们要求的是b^mw^{n-m}的系数，我们只须用二项式定理展开求得即可。

我们分别来求一下每一部分b^mw^{n-m}的系数：

1.$(b^{\overline{(n,i)}}+w^{\overline{(n,i)}})^{(n,i)}$：

若$\frac{n}{(n,i)}$不整除 m，显然b^mw^{n-m}的系数为 0，否则为$C_{(n,i)}^{m\times\overline{\frac{(n,i)}{n}}}$

2. $(b+w)(b^2+w^2)^{\overline{\frac{n-1}{2}}}$：

考虑这部分时 n 为奇数。

若 m 为偶数，则b^mw^{n-m}的系数为$C_{\overline{\frac{n-1}{2}}}^{\overline{\frac{m}{2}}}$。

若 m 为奇数，则b^mw^{n-m}的系数为$C_{\overline{\frac{n-1}{2}}}^{\overline{\frac{n-m}{2}}}$。

3. $(b+w)^2(b^2+w^2)^{\frac{n}{2}-1}$：

考虑这部分时 n 为偶数。

若 m 为偶数，则$b^m w^{n-m}$的系数为$C_{\frac{n}{2}-1}^{\frac{m}{2}-1}+C_{\frac{n}{2}-1}^{\frac{m}{2}}$。

若 m 为奇数，则$b^m w^{n-m}$的系数为$2\times C_{\frac{n}{2}-1}^{\frac{m-1}{2}}$。

4. $(b^2+w^2)^{\frac{n}{2}}$：

考虑这部分时 n 为偶数。

若 m 为偶数，则$b^m w^{n-m}$的系数为$2\times C_{\frac{n}{2}}^{\frac{m}{2}}$。

若 m 为奇数，则$b^m w^{n-m}$的系数显然为0。

这样我们将答案表示成了若干个组合数的和的形式。

我们只需要把这些组合数算出即可，当 n,m 比较大的时候需要使用高精度。

[小结]

本题是应用母函数形式的 Pólya 定理的典型题目，展示了母函数形式的 Pólya 定理将普通的 Pólya 定理不能得到的更具体的方案数计算出来。本题另一个比较重要的地方就是题目刚开始把平移这种情况排除，没有这种想法本题可能就无法下手了。这告诉我们做题时要先深入分析题目再下手做。

§3. 总结

本文从母函数的一些定义入手，介绍母函数的基本知识，并举例说明了母函数在信息学竞赛中的应用。读到这里，你会发现母函数的功能是强大的，在信息学竞赛中能帮助我们有

效的优化算法，解决问题。要用好母函数，要有深厚的数学功底，并有良好的分析问题的能力，能将问题转化至母函数的模型上。母函数本身思想的精髓在于将序列转化成函数，这种转化思想也值得我们在别的领域应用。

参考文献

[1] 吴文虎，王建德.《信息学奥林匹克竞赛指导—组合数学的算法与程序设计》.清华大学出版社，2002

[2] 刘汝佳，黄亮.《算法艺术与信息学竞赛》.清华大学出版社，2004

[3] Ronald L. Graham，Donald E. Knuth，Oren Patashnik.《Concrete Mathematics—A Foundation for Computer Science (2nd Edition)》. Addison-Wesley Professional，1994

[4] 卢开澄.《组合数学（第三版）》.清华大学出版社，2002

[5] 周源. NOI 冬令营讲稿. 2008

[6] 雷环中，金恺. IOI 中国国家集训队难题讨论活动解题报告，2003

附录

例 2. 原题

Problem F: Chocolate

(input file: chocolate.in)

In 2100, ACM chocolate will be one of the favorite foods in the world.

“Green, orange, brown, red…”, colorful sugar-coated shell maybe is the most attractive feature of ACM chocolate. How many colors have you ever seen? Nowadays, it’s said that the ACM chooses from a palette of twenty-four colors to paint their delicious candy bits.

One day, Sandy played a game on a big package of ACM chocolates which contains five colors (green, orange, brown, red and yellow). Each time he took one chocolate from the package and placed it on the table. If there were two chocolates of the same color on the table, he ate both of them. He found a quite interesting thing that in most of the time there were always 2 or 3 chocolates on the table.

Now, here comes the problem, if there are *C* colors of ACM chocolates in the package (colors are distributed evenly), after *N* chocolates are taken from the package, what’s the probability that there is exactly *M* chocolates on the table? Would you please write a program to figure it out?

Input

The input file for this problem contains several test cases, one per line.

For each case, there are three non-negative integers: *C* (*C* <= 100), *N* and *M* (*N*, *M* <= 1000000).

The input is terminated by a line containing a single zero.

Output

The output should be one real number per line, shows the probability for each case, round to three decimal places.

Sample input	Output for the sample input
5 100 2 0	0.625

例 3. 原题

Task: SWE

Sweets

Day 1. Source file swe.*

Available memory: 64MB. Maximum running time: 0.1s

John has got n jars with candies. Each of the jars contains a different kind of candies (i.e. candies from the same jar are of the same kind, and candies from different jars are of different kinds). The i-th jar contains mi candies. John has decided to eat some of his candies. He would like to eat at least a of them but no more than b. The problem is that John can’t decide how many candies and of what kinds he would like to eat. In how many ways can he do it?

Task

Your task is to write a program that:

·reads from the standard input the amount of candies in each of the jar, and integers a and b,

·determines the number of ways John can choose the candies he will eat (satisfying the above conditions),

·writes the result to the standard output

Input

The first line of input contains three integers: n, a and b, separated by single spaces (1≤n≤10, 1≤a≤b≤10000000). Each of the following n lines contains one integer. Line i+1 contains integer mi —the amount of candies in the i-th jar (0≤mi≤1000000).

Output

Let k be the number of different ways John can choose the candies to be eaten. The first and only line of output should contain one integer: k mod 2004 (i.e. the remainder of k divided by 2004).

Example

For the input data:

2 1 3

3

5

the correct result is:

9

John can choose candies in the following ways:

(1,0), (2,0), (3,0), (0,1), (0,2), (0,3), (1,1), (1,2), (2,1)