# 生成函数的运算与应用

金策

杭州学军中学

# 一般生成函数 (Ordinary Generating Function)

考虑一类组合对象组成的集合 $A$，其中，

# 一般生成函数 (Ordinary Generating Function)

考虑一类组合对象组成的集合 $A$，其中，

# 一般生成函数 (Ordinary Generating Function)

考虑一类组合对象组成的集合 $A$，其中，

- 每个元素 $a \in A$ 都被定义了“大小” $|a|$，它是一个非负整数。

# 一般生成函数 (Ordinary Generating Function)

考虑一类组合对象组成的集合 $A$，其中，

- 每个元素 $a \in A$ 都被定义了“大小” $|a|$，它是一个非负整数。
- 对于给定的 $n$，大小为 $n$ 的元素的数量是有限的，记作 $A_n$。

## 一般生成函数 (Ordinary Generating Function)

考虑一类组合对象组成的集合 $A$，其中，

- 每个元素 $a \in A$ 都被定义了“大小” $|a|$，它是一个非负整数。
- 对于给定的 $n$，大小为 $n$ 的元素的数量是有限的，记作 $A_n$。

例：$A$ 是全体 01 串组成的集合，一个 01 串的大小被定义为它的长度，则 $A_n = 2^n$。

# 一般生成函数 (Ordinary Generating Function)

我们定义

$$A(x) = \sum_{n \geq 0} A_n x^n$$

为 $A$ 的一般生成函数 (OGF)。

# 一般生成函数 (Ordinary Generating Function)

我们定义

$$A(x)=\sum_{n\geq 0}A_nx^n$$

为 $A$ 的一般生成函数 (OGF)。

例：当 $A$ 是全体 01 串时，

$$\begin{aligned}A(x)&=1+2x+4x^2+8x^3+\cdots\\&=\frac{1}{1-2x}\end{aligned}$$

# 一般生成函数 (Ordinary Generating Function)

我们定义

$$A(x) = \sum_{n\geq 0} A_n x^n$$

为 $A$ 的一般生成函数 (OGF)。

例：当 $A$ 是全体 01 串时，

$$\begin{aligned} A(x) &= 1 + 2x + 4x^2 + 8x^3 + \cdots \\ &= \frac{1}{1-2x} \end{aligned}$$

形式幂级数，无需考虑级数在何时收敛。

# 一般生成函数 (Ordinary Generating Function)

我们定义

$$A(x) = \sum_{n\geq 0} A_n x^n$$

为 $A$ 的一般生成函数 (OGF)。

例：当 $A$ 是全体 01 串时，

$$\begin{aligned} A(x) &= 1 + 2x + 4x^2 + 8x^3 + \cdots \\ &= \frac{1}{1-2x} \end{aligned}$$

形式幂级数，无需考虑级数在何时收敛。
实际做题时往往只需处理级数的前 $n$ 项系数 (模一个大质数)。

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）
  显然 $C(x) = A(x) + B(x)$。

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）
  显然 $C(x) = A(x) + B(x)$。$O(n)$ 计算。

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）
  显然 $C(x) = A(x) + B(x)$。$O(n)$ 计算。
- 定义 $D$ 为 $A$ 和 $B$ 的笛卡尔积，

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）
  显然 $C(x) = A(x) + B(x)$。$O(n)$ 计算。
- 定义 $D$ 为 $A$ 和 $B$ 的笛卡尔积，
  $D$ 的每个元素 $d$ 都是 $A$ 的某元素 $a$ 与 $B$ 的某元素 $b$ 拼成的二元组 $(a, b)$，其大小 $|d|$ 定义为 $|a| + |b|$，

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）
  显然 $C(x) = A(x) + B(x)$。$O(n)$ 计算。
- 定义 $D$ 为 $A$ 和 $B$ 的笛卡尔积，
  $D$ 的每个元素 $d$ 都是 $A$ 的某元素 $a$ 与 $B$ 的某元素 $b$ 拼成的二元组 $(a, b)$，其大小 $|d|$ 定义为 $|a| + |b|$，
  显然 $D(x) = A(x)B(x)$。

# 最基本的运算

有两类组合对象 $A$ 和 $B$，

- 定义 $C$ 为 $A$ 和 $B$ 的并集，（$c$ 是 $a$ 或 $b$）
  显然 $C(x) = A(x) + B(x)$。$O(n)$ 计算。
- 定义 $D$ 为 $A$ 和 $B$ 的笛卡尔积，
  $D$ 的每个元素 $d$ 都是 $A$ 的某元素 $a$ 与 $B$ 的某元素 $b$ 拼成的二元组 $(a, b)$，其大小 $|d|$ 定义为 $|a| + |b|$，
  显然 $D(x) = A(x)B(x)$。FFT 乘法 $O(n \log n)$。

# 组成序列

# 组成序列

有一类组合对象 $A$，

# 组成序列

有一类组合对象 $A$，
定义 $\mathcal{SEQ}(A)$ 是由 $A$ 的元素排成的序列组成的集合，

# 组成序列

有一类组合对象 $A$，

定义 $\mathcal{SEQ}(A)$ 是由 $A$ 的元素排成的序列组成的集合，一个序列的大小定义为其元素大小总和。

# 组成序列

有一类组合对象 $A$，
定义 $\mathcal{SEQ}(A)$ 是由 $A$ 的元素排成的序列组成的集合，一个序列的大小定义为其元素大小总和。

- 例 1：$A = \{"0", "1"\}$，则 $\mathcal{SEQ}(A) = \{01$ 串$\}$。

# 组成序列

有一类组合对象 $A$，

定义 $\mathcal{SEQ}(A)$ 是由 $A$ 的元素排成的序列组成的集合，一个序列的大小定义为其元素大小总和。

- 例 1：$A = \{"0", "1"\}$，则 $\mathcal{SEQ}(A) = \{01 \text{ 串}\}$。
- 例 2：正整数集 $N = \{1, 2, 3, \cdots\}$，元素的大小定义为它的数值，则 $\mathcal{SEQ}(N) = \{\text{正整数有序拆分}\} = \{0, 1, 1+1, 2, 1+1+1, 1+2, 2+1, 3, \cdots\}$。

# 组成序列

规定 $A$ 中不含大小为 0 的元素，则

$$\begin{aligned}\mathcal{SEQ}(A) &= 1 + A + A \cdot A + A^3 + \cdots \\ &= \frac{1}{1-A}\end{aligned}$$

# 组成序列

规定 $A$ 中不含大小为 0 的元素，则

$$\begin{aligned} \mathcal{SEQ}(A) &= 1 + A + A \cdot A + A^3 + \cdots \\ &= \frac{1}{1-A} \end{aligned}$$

- 例 1：$A = \{"0","1"\}$，$A(x) = 2x$，

# 组成序列

规定 $A$ 中不含大小为 0 的元素，则

$$\begin{aligned} \mathcal{SEQ}(A) &= 1 + A + A \cdot A + A^3 + \cdots \\ &= \frac{1}{1-A} \end{aligned}$$

- 例 1：$A = \{"0", "1"\}$，$A(x) = 2x$，$\mathcal{SEQ}(A) = 1/(1-2x)$。

# 组成序列

规定 $A$ 中不含大小为 0 的元素，则

$$\begin{aligned}\mathcal{SEQ}(A) &= 1 + A + A \cdot A + A^3 + \cdots \\ &= \frac{1}{1 - A}\end{aligned}$$

- 例 1：$A = \{"0", "1"\}$，$A(x) = 2x$，$\mathcal{SEQ}(A) = 1/(1 - 2x)$。
- 例 2：正整数集 $N = \{1, 2, 3, \cdots\}$，

# 组成序列

规定 $A$ 中不含大小为 0 的元素，则

$$\begin{aligned}\mathcal{SEQ}(A) &= 1 + A + A\cdot A + A^3 + \cdots \\ &= \frac{1}{1-A}\end{aligned}$$

- 例 1： $A = \{"0", "1"\}$， $A(x) = 2x$， $\mathcal{SEQ}(A) = 1/(1-2x)$。
- 例 2： 正整数集 $N = \{1, 2, 3, \cdots\}$，

$$N(x) = x + x^2 + \cdots = \frac{x}{1-x}$$

$$\mathcal{SEQ}(N) = \frac{1}{1-N(x)} = 1 + \frac{x}{1-2x} = 1 + x + 2x^2 + 4x^3 + \cdots$$

# 求乘法逆元

# 求乘法逆元

已知 $f(x)$，可以用 $O(n\log n)$ 时间计算 $1/f(x)$。（模 $x^n$ 意义下）
条件：$f(x)$ 的常数项存在逆元。

# 求乘法逆元

已知 $f(x)$，可以用 $O(n\log n)$ 时间计算 $1/f(x)$。（模 $x^n$ 意义下）
条件：$f(x)$ 的常数项存在逆元。
做法参见去年营员交流余行江、彭雨翔大神的《多项式除法》。(orz!)
相当于牛顿迭代。

# 带标号的对象

有时我们需要考虑带标号的组合对象，比如标号图。

# 带标号的对象

有时我们需要考虑带标号的组合对象，比如标号图。
$n$ 个点的标号图，顶点的标号恰好为 $1, 2, \cdots, n$。

# 带标号对象的拼接

# 带标号对象的拼接

将两个对象 $a, b$ 拼接起来，$|a| = n, |b| = m$

# 带标号对象的拼接

将两个对象 $a, b$ 拼接起来，$|a| = n, |b| = m$
无标号时，只有一种方法；

# 带标号对象的拼接

将两个对象 $a, b$ 拼接起来，$|a| = n, |b| = m$
无标号时，只有一种方法；
带标号时，规定拼接时需保持标号的原有相对顺序，有

$$\binom{n+m}{n} = \frac{(n+m)!}{n!m!}$$

种方法。

# 带标号对象的拼接

将两个对象 $a, b$ 拼接起来，$|a| = n, |b| = m$
无标号时，只有一种方法；
带标号时，规定拼接时需保持标号的原有相对顺序，有

$$\binom{n+m}{n} = \frac{(n+m)!}{n!m!}$$

种方法。

比如，将 213 和 21 合并，有
435,21  425,31  325,41  324,51  415,32
315,42  314,52  215,43  214,53  213,54
这 10 种方法。

# 指数生成函数 (Exponential Generating Function)

对于带标号组合对象组成的集合 A，定义

$$A(x)=\sum_{n\geq 0}A_n\frac{x^n}{n!}$$

为 $A$ 的指数生成函数 (EGF)。

# 最基本的运算

# 最基本的运算

- 并集：$C(x) = A(x) + B(x)$。

# 最基本的运算

- 并集：$C(x) = A(x) + B(x)$。
- $a$ 和 $b$ 拼接：$D(x) = A(x) \cdot B(x)$。

# 最基本的运算

- 并集：$C(x) = A(x) + B(x)$。
- $a$ 和 $b$ 拼接：$D(x) = A(x) \cdot B(x)$。
  原因：
  $$D_n = \sum_{i+j=n} A_i B_j \frac{(i+j)!}{i!j!}$$
  比较一下系数即可。

# 集合的 EGF

# 集合的 EGF

- 序列的 EGF，

# 集合的 EGF

- 序列的 EGF,
  顺序是确定的,

# 集合的 EGF

- 序列的 EGF，
  顺序是确定的，

$$\mathcal{SEQ}(A) = \sum_{i \geq 0} A^i = \frac{1}{1 - A}$$

# 集合的 EGF

- 序列的 EGF，
  顺序是确定的，

$$\mathcal{SEQ}(A) = \sum_{i \geq 0} A^i = \frac{1}{1-A}$$

- 集合的 EGF，

# 集合的 EGF

- 序列的 EGF，
  顺序是确定的，

$$\mathcal{SEQ}(A)=\sum_{i\geq 0}A^i=\frac{1}{1-A}$$

- 集合的 EGF，
  顺序是无关紧要的，

# 集合的 EGF

- 序列的 EGF，
  顺序是确定的，

$$\mathcal{SEQ}(A) = \sum_{i\geq 0} A^i = \frac{1}{1-A}$$

- 集合的 EGF，
  顺序是无关紧要的，

$$\mathcal{SET}(A) = \sum_{i\geq 0} \frac{A^i}{i!} = e^A$$

# 集合的 EGF

例：一个简单无向图是若干连通分量组成的集合。

# 集合的 EGF

例：一个简单无向图是若干连通分量组成的集合。

$G$ 是简单无向图的 EGF，

$$G(x) = \sum_{n\geq 0} 2^{\binom{n}{2}} \frac{x^n}{n!}$$

# 集合的 EGF

例：一个简单无向图是若干连通分量组成的集合。

$G$ 是简单无向图的 EGF，

$$G(x) = \sum_{n\geq 0} 2^{\binom{n}{2}} \frac{x^n}{n!}$$

$C$ 是简单连通图的 EGF，

$$C(x) = \sum_{n\geq 1} C_n \frac{x^n}{n!}$$

# 集合的 EGF

例：一个简单无向图是若干连通分量组成的集合。

$G$ 是简单无向图的 EGF，

$$G(x) = \sum_{n\geq 0} 2^{\binom{n}{2}} \frac{x^n}{n!}$$

$C$ 是简单连通图的 EGF，

$$C(x) = \sum_{n\geq 1} C_n \frac{x^n}{n!}$$

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

# 集合的 EGF

例：一个简单无向图是若干连通分量组成的集合。
$G$ 是简单无向图的 EGF，

$$G(x) = \sum_{n \geq 0} 2^{\binom{n}{2}} \frac{x^n}{n!}$$

$C$ 是简单连通图的 EGF，

$$C(x) = \sum_{n \geq 1} C_n \frac{x^n}{n!}$$

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

从而

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

# 计算 ln

给定 $f(x)$，如何求 $\ln f(x)$？

# 计算 ln

给定 $f(x)$，如何求 $\ln f(x)$？

令

$$g(x) = \ln f(x)$$

# 计算 ln

给定 $f(x)$，如何求 $\ln f(x)$?
令
$$g(x) = \ln f(x)$$
则
$$g'(x) = \frac{f'(x)}{f(x)}$$

# 计算 ln

给定 $f(x)$，如何求 $\ln f(x)$?

令

$$g(x) = \ln f(x)$$

则

$$g'(x) = \frac{f'(x)}{f(x)}$$

逆元我们已经会做了。求导和积分都是 $O(n)$ 的。

# 计算 ln

给定 $f(x)$，如何求 $\ln f(x)$?
令

$$g(x) = \ln f(x)$$

则

$$g'(x) = \frac{f'(x)}{f(x)}$$

逆元我们已经会做了。求导和积分都是 $O(n)$ 的。
总时间 $O(n \log n)$。

# 计算 exp

给定 $f(x)$，如何求 $e^{f(x)}$？

# 计算 exp

给定 $f(x)$，如何求 $e^{f(x)}$？
（根据 $\left(e^{f(x)}\right)' = f'(x)e^{f(x)}$ 进行分治 FFT，复杂度 $O(n\log^2 n)$）

# 计算 exp

给定 $f(x)$，如何求 $e^{f(x)}$？
（根据 $\left(e^{f(x)}\right)' = f'(x)e^{f(x)}$ 进行分治 FFT，复杂度 $O(n\log^2 n)$）
用牛顿迭代可做到 $O(n\log n)$。

# 牛顿迭代

有一个关于 $f$ 的方程 $g(f)=0$，其中 $f$ 是一个未知的形式幂级数。

# 牛顿迭代

有一个关于 $f$ 的方程 $g(f)=0$，其中 $f$ 是一个未知的形式幂级数。假如我们已知 $f$ 的前 $n$ 项 $f_0$，即

$$f \equiv f_0 \pmod{x^n}$$

# 牛顿迭代

有一个关于 $f$ 的方程 $g(f)=0$，其中 $f$ 是一个未知的形式幂级数。假如我们已知 $f$ 的前 $n$ 项 $f_0$，即

$$f \equiv f_0 \pmod{x^n}$$

根据泰勒展开，有

$$\begin{aligned} 0 = g(f) &= g(f_0) + g'(f_0)(f - f_0) + \frac{g''(f_0)}{2}(f - f_0)^2 + \cdots \\ &\equiv g(f_0) + g'(f_0)(f - f_0) \pmod{x^{2n}} \end{aligned}$$

即

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

# 计算 exp

# 计算 exp

假设我们要求 $f(x) = e^{A(x)}$,

# 计算 exp

假设我们要求 $f(x) = e^{A(x)}$，

$$g(f) = \ln(f) - A = 0$$

# 计算 exp

假设我们要求 $f(x) = e^{A(x)}$，

$$g(f) = \ln(f) - A = 0$$

根据刚才的式子

$$\begin{aligned} f &= f_0 - \frac{g(f_0)}{g'(f_0)} \\ &= f_0 - \frac{\ln(f_0) - A}{1/f_0} \\ &= f_0(1 - \ln(f_0) + A) \end{aligned}$$

# 计算 exp

假设我们要求 $f(x) = e^{A(x)}$，

$$g(f) = \ln(f) - A = 0$$

根据刚才的式子

$$\begin{aligned} f &= f_0 - \frac{g(f_0)}{g'(f_0)} \\ &= f_0 - \frac{\ln(f_0) - A}{1/f_0} \\ &= f_0(1 - \ln(f_0) + A) \end{aligned}$$

按此迭代，复杂度 $T(n) = T(n/2) + O(n \log n) = O(n \log n)$。

# ln 和 exp

# ln 和 exp

给定 $f(x)$，如何求出 $f(x)^k \bmod x^n$？
（DFT 后对每个点值求 $k$ 次幂的做法是错的，因为 mod $x^n$ 后多项式的点值会变化）

# ln 和 exp

给定 $f(x)$，如何求出 $f(x)^k \bmod x^n$？
（DFT 后对每个点值求 $k$ 次幂的做法是错的，因为 mod $x^n$ 后多项式的点值会变化）

- 倍增快速幂，每次 FFT 乘法 $O(n\log n)$，总时间 $O(n\log n\log k)$。

# ln 和 exp

给定 $f(x)$，如何求出 $f(x)^k \bmod x^n$？
（DFT 后对每个点值求 $k$ 次幂的做法是错的，因为 mod $x^n$ 后多项式的点值会变化）

- 倍增快速幂，每次 FFT 乘法 $O(n\log n)$，总时间 $O(n\log n\log k)$。
-
$$f(x)^k = \exp\left(\ln(f(x)^k)\right) = \exp\left(k\ln f(x)\right)$$

  时间 $O(n\log n)$。

# ln 和 exp

给定 $f(x)$，如何求出 $f(x)^k \bmod x^n$？
（DFT 后对每个点值求 $k$ 次幂的做法是错的，因为 mod $x^n$ 后多项式的点值会变化）

- 倍增快速幂，每次 FFT 乘法 $O(n\log n)$，总时间 $O(n\log n\log k)$。
- $$f(x)^k = \exp\left(\ln(f(x)^k)\right) = \exp\left(k\ln f(x)\right)$$
  时间 $O(n\log n)$。
  注意：代入前要先将 $f(x)$ 除以 $cx^d$，使其常数项变成 1

# ln 和 exp

给定 $f(x)$，如何求出 $f(x)^k \bmod x^n$？
（DFT 后对每个点值求 $k$ 次幂的做法是错的，因为 mod $x^n$ 后多项式的点值会变化）

- 倍增快速幂，每次 FFT 乘法 $O(n\log n)$，总时间 $O(n\log n\log k)$。
-
$$f(x)^k = \exp\left(\ln(f(x)^k)\right) = \exp\left(k\ln f(x)\right)$$
  时间 $O(n\log n)$。
  注意：代入前要先将 $f(x)$ 除以 $cx^d$，使其常数项变成 1
- 也可以用来求 $k$ 次方根。

# ln 和 exp

给定 $f(x)$，如何求出 $f(x)^k \bmod x^n$？
（DFT 后对每个点值求 $k$ 次幂的做法是错的，因为 mod $x^n$ 后多项式的点值会变化）

- 倍增快速幂，每次 FFT 乘法 $O(n\log n)$，总时间 $O(n\log n\log k)$。
- $$f(x)^k = \exp\left(\ln(f(x)^k)\right) = \exp\left(k\ln f(x)\right)$$
  时间 $O(n\log n)$。
  注意：代入前要先将 $f(x)$ 除以 $cx^d$，使其常数项变成 1
- 也可以用来求 $k$ 次方根。
  代入前同样要注意调整常数项。显然 $f(x)$ 的最低次幂指数要保证为 $k$ 的倍数。

# 关于置换的计数

# 关于置换的计数

一个置换是若干个轮换组成的集合。

# 关于置换的计数

一个置换是若干个轮换组成的集合。

$k$-轮换的数量有 $(k-1)!$ 个，对应的 EGF 为

$$\frac{(k-1)!}{k!}x^k = \frac{x^k}{k}$$

# 关于置换的计数

一个置换是若干个轮换组成的集合。
$k$-轮换的数量有 $(k-1)!$ 个，对应的 EGF 为

$$\frac{(k-1)!}{k!}x^k = \frac{x^k}{k}$$

所有轮换的 EGF 为

$$\sum_{k\geq 1}\frac{x^k}{k} = -\ln(1-x)$$

## 关于置换的计数

一个置换是若干个轮换组成的集合。

$k$-轮换的数量有 $(k-1)!$ 个，对应的 EGF 为

$$\frac{(k-1)!}{k!}x^k = \frac{x^k}{k}$$

所有轮换的 EGF 为

$$\sum_{k\geq 1}\frac{x^k}{k} = -\ln(1-x)$$

于是所有置换的 EGF 为

$$\exp\left(-\ln(1-x)\right) = \frac{1}{1-x}$$

，这符合 $n$-置换数量为 $n!$ 的事实。

# 关于置换的计数

如果限制每个轮换的大小在集合 $S$ 内呢？

# 关于置换的计数

如果限制每个轮换的大小在集合 $S$ 内呢？
很简单，

$$\exp\left(\sum_{k\in S}\frac{x^k}{k}\right)$$

# 背包计数

（无标号的集合计数）

# 背包计数

（无标号的集合计数）

- 版本 1：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个。

# 背包计数

（无标号的集合计数）

- 版本 1：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个。
- 版本 2：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品只有 1 个。

# 背包计数

（无标号的集合计数）

- 版本 1：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个。
- 版本 2：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品只有 1 个。
- 版本 3：有 $n$ 种物品，第 $i$ 种物品体积为 $i$。第 $i$ 种物品有 $a_i$ 个。

# 背包计数

（无标号的集合计数）

- 版本 1：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个。
- 版本 2：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品只有 1 个。
- 版本 3：有 $n$ 种物品，第 $i$ 种物品体积为 $i$。第 $i$ 种物品有 $a_i$ 个。

对于所有 $m \in \{1, 2, \cdots, n\}$，回答选取物品恰好总体积为 $m$ 的方案数。

# 背包计数

（无标号的集合计数）

- 版本 1：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个。
- 版本 2：有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品只有 1 个。
- 版本 3：有 $n$ 种物品，第 $i$ 种物品体积为 $i$。第 $i$ 种物品有 $a_i$ 个。

对于所有 $m \in \{1, 2, \cdots, n\}$，回答选取物品恰好总体积为 $m$ 的方案数。

均可以在 $O(n \log n)$ 时间求解。

# 背包计数

下面以版本 1 为例。其他版本处理方法类似。

(有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个)

# 背包计数

下面以版本 1 为例。其他版本处理方法类似。

(有 $\sum_{1\le i\le n} a_i$ 种物品，其中体积为 $i$ 的物品有 $a_i$ 种。每种物品有无限个)

$$\begin{aligned}\prod_{1\le i\le n}(1+x^i+x^{2i}+\cdots)^{a_i} &= \prod_{1\le i\le n}(\frac{1}{1-x^i})^{a_i}\\ &=\exp\left(-\sum_{1\le i\le n} a_i\ln(1-x^i)\right)\\ &=\exp\left(\sum_{1\le i\le n} a_i\sum_{j\ge 1}\frac{x^{ij}}{j}\right)\\ &=\exp\left(\sum_{j\ge 1}\frac{1}{j}A(x^j)\right)\end{aligned}$$

# 背包计数

$$\exp\left(\sum_{j\geq 1}\frac{1}{j}A(x^j)\right)$$

# 背包计数

$$\exp\left(\sum_{j\geq 1}\frac{1}{j}A(x^j)\right)$$

$A(x^j)$ 中只有 $n/j$ 项是有用的，

# 背包计数

$$\exp\left(\sum_{j\geq 1}\frac{1}{j}A(x^j)\right)$$

$A(x^j)$ 中只有 $n/j$ 项是有用的，
可以用 $n+\frac{n}{2}+\frac{n}{3}+\cdots+1=O(n\log n)$ 时间求和。

# 背包计数

$$\exp\left(\sum_{j\ge 1}\frac{1}{j}A(x^j)\right)$$

$A(x^j)$ 中只有 $n/j$ 项是有用的，
可以用 $n+\frac{n}{2}+\frac{n}{3}+\cdots+1=O(n\log n)$ 时间求和。
再做一次 exp，总复杂度 $O(n\log n)$。

# 复合与复合逆

$$A(x) = \sum_{i\geq 1} a_i x^i, B(x) = \sum_{i\geq 1} b_i x^i$$

# 复合与复合逆

$$A(x) = \sum_{i \geq 1} a_i x^i, B(x) = \sum_{i \geq 1} b_i x^i$$

- 复合

$$A \circ B(x) = A\left(B(x)\right) = \sum_{i \geq 1} a_i \left(B(x)\right)^i$$

# 复合与复合逆

$$A(x) = \sum_{i \geq 1} a_i x^i, B(x) = \sum_{i \geq 1} b_i x^i$$

- 复合

$$A \circ B(x) = A\left(B(x)\right) = \sum_{i \geq 1} a_i \left(B(x)\right)^i$$

- 复合逆（反函数）

$$f\left(g(x)\right) = g\left(f(x)\right) = x$$

$f$ 和 $g$ 互为反函数。显然 $f, g$ 的一次项系数互为乘法逆元。

# 复合与复合逆

若能够计算复合，就可以用牛顿迭代求出复合逆（回忆刚才求 exp 的过程）。

## 复合与复合逆

若能够计算复合，就可以用牛顿迭代求出复合逆（回忆刚才求 exp 的过程）。

暴力：$O(n^3)$

# 复合与复合逆

若能够计算复合，就可以用牛顿迭代求出复合逆（回忆刚才求 exp 的过程）。

暴力：$O(n^3)$

FFT 暴力：$O(n^2 \log n)$

# 复合与复合逆

若能够计算复合，就可以用牛顿迭代求出复合逆（回忆刚才求 exp 的过程）。

暴力：$O(n^3)$

FFT 暴力：$O(n^2 \log n)$

小步大步 +FFT：$O(n^2)$

# 复合与复合逆

若能够计算复合，就可以用牛顿迭代求出复合逆（回忆刚才求 exp 的过程）。

暴力：$O(n^3)$

FFT 暴力：$O(n^2 \log n)$

小步大步 +FFT：$O(n^2)$

Fast Algorithms for Manipulating Formal Power Series (Brent & Kung) 给出了 $O\left((n\log n)^{3/2}\right)$ 求复合、复合逆的算法。

# 复合与复合逆

若能够计算复合，就可以用牛顿迭代求出复合逆（回忆刚才求 exp 的过程）。

暴力：$O(n^3)$

FFT 暴力：$O(n^2 \log n)$

小步大步 +FFT：$O(n^2)$

Fast Algorithms for Manipulating Formal Power Series (Brent & Kung) 给出了 $O\left((n \log n)^{3/2}\right)$ 求复合、复合逆的算法。

有一点点长，这里就不介绍啦。

# 拉格朗日反演 (Lagrange Inversion)

# 拉格朗日反演 (Lagrange Inversion)

若 $f$ 是 $g$ 的复合逆，则

$$[x^n]g(x) = \frac{1}{n}[x^{n-1}]\left(\frac{x}{f(x)}\right)^n$$

# 拉格朗日反演 (Lagrange Inversion)

若 $f$ 是 $g$ 的复合逆，则

$$[x^n]g(x) = \frac{1}{n}[x^{n-1}]\left(\frac{x}{f(x)}\right)^n$$

推广形式 (取 $h(x) = x$ 即为上式):

$$[x^n]h\left(g(x)\right) = \frac{1}{n}[x^{n-1}]h'(x)\left(\frac{x}{f(x)}\right)^n$$

# 拉格朗日反演 (Lagrange Inversion)

若 $f$ 是 $g$ 的复合逆，则

$$[x^n]g(x) = \frac{1}{n}[x^{n-1}]\left(\frac{x}{f(x)}\right)^n$$

推广形式 (取 $h(x) = x$ 即为上式):

$$[x^n]h\left(g(x)\right) = \frac{1}{n}[x^{n-1}]h'(x)\left(\frac{x}{f(x)}\right)^n$$

证明过程就不介绍啦。

# 一个应用

$$[x^n]g(x) = \frac{1}{n}[x^{n-1}]\left(\frac{x}{f(x)}\right)^n$$

$g(x)$ 的反函数不容易求，但根据上式可用 $O(n\log n)$ 时间求出其 $x^n$ 项系数。

## 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 ($n \le 10^5$)
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。($1 \notin S$)

# 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 $(n \leq 10^5)$
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。$(1 \notin S)$

令这些树的生成函数为 $T$。

# 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 ($n \leq 10^5$)
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。($1 \notin S$)

令这些树的生成函数为 $T$。

一棵树是一个叶子结点，或者一个非叶节点拼上 $s(s \in S)$ 棵树组成的序列

# 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 $(n \le 10^5)$
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。$(1 \notin S)$

令这些树的生成函数为 $T$。

一棵树是一个叶子结点，或者一个非叶节点拼上 $s(s \in S)$ 棵树组成的序列

$$T = x^1 + x^0 \sum_{s \in S} T^s$$

# 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 ($n \leq 10^5$)
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。($1 \notin S$)

令这些树的生成函数为 $T$。

一棵树是一个叶子结点，或者一个非叶节点拼上 $s(s \in S)$ 棵树组成的序列

$$T = x^1 + x^0 \sum_{s \in S} T^s$$

于是 $x = f(T)$，$f$ 是一个一次项系数为 1 的多项式

## 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 ($n \leq 10^5$)
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。($1 \notin S$)

令这些树的生成函数为 $T$。

一棵树是一个叶子结点，或者一个非叶节点拼上 $s(s \in S)$ 棵树组成的序列

$$T = x^1 + x^0 \sum_{s \in S} T^s$$

于是 $x = f(T)$，$f$ 是一个一次项系数为 1 的多项式

所以 $T = g(x)$，其中 $g$ 是 $f$ 的复合逆。

# 一个应用

例题 (BZOJ3684 大朋友和多叉树)：求出满足以下性质的有根多叉树的数量（结点无标号，孩子有顺序）

- 共有 $n$ 个叶子结点 $(n \le 10^5)$
- 每个非叶结点的儿子数量 $\in$ 集合 $S$。$(1 \notin S)$

令这些树的生成函数为 $T$。

一棵树是一个叶子结点，或者一个非叶节点拼上 $s(s \in S)$ 棵树组成的序列

$$T = x^1 + x^0 \sum_{s \in S} T^s$$

于是 $x = f(T)$，$f$ 是一个一次项系数为 1 的多项式

所以 $T = g(x)$，其中 $g$ 是 $f$ 的复合逆。求出其 $n$ 次项系数即可。

# 另一个应用

# 另一个应用

一个不太好做的问题：

# 另一个应用

一个不太好做的问题：

- 给定一个多项式 $f(x)$，和一个数 $k$。

# 另一个应用

一个不太好做的问题：

- 给定一个多项式 $f(x)$，和一个数 $k$。
- 对于所有 $1 \le i \le n$，求出 $[x^k]f(x)^i$。（假定 $k = O(n)$ 吧）

# 另一个应用

一个不太好做的问题：

- 给定一个多项式 $f(x)$，和一个数 $k$。
- 对于所有 $1 \leq i \leq n$，求出 $[x^k]f(x)^i$。（假定 $k = O(n)$ 吧）

FFT 暴力：$O(n^2 \log n)$

# 另一个应用

一个不太好做的问题：

- 给定一个多项式 $f(x)$，和一个数 $k$。
- 对于所有 $1 \le i \le n$，求出 $[x^k]f(x)^i$。（假定 $k = O(n)$ 吧）

FFT 暴力：$O(n^2 \log n)$
小步大步 +FFT 暴力：$O(n^2)$

# 另一个应用

一个不太好做的问题：

- 给定一个多项式 $f(x)$，和一个数 $k$。
- 对于所有 $1 \leq i \leq n$，求出 $[x^k]f(x)^i$。（假定 $k = O(n)$ 吧）

FFT 暴力：$O(n^2 \log n)$

小步大步 +FFT 暴力：$O(n^2)$

分析一下。

# 另一个应用

所求的答案序列即为

$$1 + uf(x) + u^2 f(x)^2 + \cdots = \frac{1}{1 - uf(x)}$$

中的 $x^k$ 项系数，它是一个关于 $u$ 的多项式。

# 另一个应用

所求的答案序列即为

$$1 + uf(x) + u^2 f(x)^2 + \cdots = \frac{1}{1 - uf(x)}$$

中的 $x^k$ 项系数，它是一个关于 $u$ 的多项式。

最方便的情况，$[x^0]f(x) = 0, [x^1]f(x) \neq 0$，于是 $f(x)$ 存在复合逆 $g(x)$，则根据拉格朗日反演的推广形式

$$[x^k]\frac{1}{1 - uf(x)} = \frac{1}{k}[x^{k-1}]\frac{u}{(1 - ux)^2}\left(\frac{x}{g(x)}\right)^k$$

# 另一个应用

所求的答案序列即为

$$1 + uf(x) + u^2 f(x)^2 + \cdots = \frac{1}{1 - uf(x)}$$

中的 $x^k$ 项系数，它是一个关于 $u$ 的多项式。

最方便的情况，$[x^0]f(x) = 0, [x^1]f(x) \neq 0$，于是 $f(x)$ 存在复合逆 $g(x)$，则根据拉格朗日反演的推广形式

$$[x^k]\frac{1}{1 - uf(x)} = \frac{1}{k}[x^{k-1}]\frac{u}{(1 - ux)^2}\left(\frac{x}{g(x)}\right)^k$$

由此发现，这个问题与求解 $f(x)$ 的复合逆是可以在 $O(n \log n)$ 时间内相互转化的。

# 另一个应用

所求的答案序列即为

$$1 + uf(x) + u^2 f(x)^2 + \cdots = \frac{1}{1 - uf(x)}$$

中的 $x^k$ 项系数，它是一个关于 $u$ 的多项式。

最方便的情况，$[x^0]f(x) = 0, [x^1]f(x) \neq 0$，于是 $f(x)$ 存在复合逆 $g(x)$，则根据拉格朗日反演的推广形式

$$[x^k]\frac{1}{1 - uf(x)} = \frac{1}{k}[x^{k-1}]\frac{u}{(1 - ux)^2}\left(\frac{x}{g(x)}\right)^k$$

由此发现，这个问题与求解 $f(x)$ 的复合逆是可以在 $O(n \log n)$ 时间内相互转化的。

如果 $f(x)$ 长得不好看，不存在复合逆，还需做些处理。

# 另一个应用

# 另一个应用

- 如果 $f(x)$ 常数项为 0，但最低次项 $f_t x^t$ 的次数 $t \geq 2$，那么对 $f(x)/f_t$ 开 $t$ 次方根，表示成 $f(x) = f_t \cdot (p(x))^t$，那么 $p(x)$ 的最低次项次数为 1，继续套用上述方法。

# 另一个应用

- 如果 $f(x)$ 常数项为 0，但最低次项 $f_t x^t$ 的次数 $t \geq 2$，那么对 $f(x)/f_t$ 开 $t$ 次方根，表示成 $f(x) = f_t \cdot (p(x))^t$，那么 $p(x)$ 的最低次项次数为 1，继续套用上述方法。
- 如果 $f(x)$ 的常数项是 $c \neq 0$，那么写成 $f(x) = c + q(x)$。我们能够求出所有 $[x^k]q(x)^i$。

# 另一个应用

- 如果 $f(x)$ 常数项为 0，但最低次项 $f_t x^t$ 的次数 $t \geq 2$，那么对 $f(x)/f_t$ 开 $t$ 次方根，表示成 $f(x) = f_t \cdot (p(x))^t$，那么 $p(x)$ 的最低次项次数为 1，继续套用上述方法。
- 如果 $f(x)$ 的常数项是 $c \neq 0$，那么写成 $f(x) = c + q(x)$。我们能够求出所有 $[x^k]q(x)^i$。

$$
\begin{aligned}
[x^k]\left(c+q(x)\right)^i &= [x^k]\sum_{0\leq j\leq i}\binom{i}{j}c^j q(x)^{i-j} \\
&= i!\sum_{0\leq j\leq i}\frac{c^j}{j!}\frac{[x^k]q(x)^{i-j}}{(i-j)!}
\end{aligned}
$$

# 另一个应用

- 如果 $f(x)$ 常数项为 0，但最低次项 $f_t x^t$ 的次数 $t \geq 2$，那么对 $f(x)/f_t$ 开 $t$ 次方根，表示成 $f(x) = f_t \cdot (p(x))^t$，那么 $p(x)$ 的最低次项次数为 1，继续套用上述方法。
- 如果 $f(x)$ 的常数项是 $c \neq 0$，那么写成 $f(x) = c + q(x)$。我们能够求出所有 $[x^k]q(x)^i$。

$$
\begin{aligned}
[x^k]\left(c + q(x)\right)^i &= [x^k] \sum_{0 \leq j \leq i} \binom{i}{j} c^j q(x)^{i-j} \\
&= i! \sum_{0 \leq j \leq i} \frac{c^j}{j!} \frac{[x^k]q(x)^{i-j}}{(i-j)!}
\end{aligned}
$$

再 FFT 求一次卷积即可。

# 另一个应用

只要能求出 $f(x)$ 的复合逆，就可以再用 $O(n\log n)$ 时间解决上述问题。一般情况下求复合逆我只会 $O\left((n\log n)^{3/2}\right)$ 的做法。但某些特殊情况下 $f(x)$ 的复合逆比较容易求得。

# 另一个应用

只要能求出 $f(x)$ 的复合逆，就可以再用 $O(n\log n)$ 时间解决上述问题。一般情况下求复合逆我只会 $O\left((n\log n)^{3/2}\right)$ 的做法。但某些特殊情况下 $f(x)$ 的复合逆比较容易求得。

例：给定 $n$，求出所有 $\begin{bmatrix} n \\ k \end{bmatrix}, 0 \le k \le n$。（第一类斯特林数）

# 另一个应用

只要能求出 $f(x)$ 的复合逆，就可以再用 $O(n\log n)$ 时间解决上述问题。一般情况下求复合逆我只会 $O\left((n\log n)^{3/2}\right)$ 的做法。但某些特殊情况下 $f(x)$ 的复合逆比较容易求得。

例：给定 $n$，求出所有 $\begin{bmatrix} n \\ k \end{bmatrix}, 0 \le k \le n$。（第一类斯特林数）

传统做法：利用

$$\sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} x^k = x(x+1)(x+2)\cdots(x+n-1)$$

分治 FFT，复杂度 $O(n\log^2 n)$。

# 另一个应用

只要能求出 $f(x)$ 的复合逆，就可以再用 $O(n\log n)$ 时间解决上述问题。一般情况下求复合逆我只会 $O\left((n\log n)^{3/2}\right)$ 的做法。但某些特殊情况下 $f(x)$ 的复合逆比较容易求得。

例：给定 $n$，求出所有 $\begin{bmatrix} n \\ k \end{bmatrix}, 0 \le k \le n$。（第一类斯特林数）

传统做法：利用

$$\sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} x^k = x(x+1)(x+2)\cdots(x+n-1)$$

分治 FFT，复杂度 $O(n\log^2 n)$。

另一个做法：

$$\begin{bmatrix} n \\ k \end{bmatrix} = \frac{n!}{k!}[z^n]\left(-\ln(1-x)\right)^k$$

套用刚才的方法，复杂度 $O(n\log n)$。由于某些原因跑不过 $O(n\log^2 n)$。

# 感谢

# 感谢

感谢 CCF 给我提供这次交流的机会。

# 感谢

感谢 CCF 给我提供这次交流的机会。
感谢教练徐先友老师给予我信心、能力、技巧、毅力。

## 感谢

感谢 CCF 给我提供这次交流的机会。
感谢教练徐先友老师给予我信心、能力、技巧、毅力。
感谢吕凯风、毛啸、彭雨翔等同学提出的宝贵意见。

# 感谢

感谢 CCF 给我提供这次交流的机会。
感谢教练徐先友老师给予我信心、能力、技巧、毅力。
感谢吕凯风、毛啸、彭雨翔等同学提出的宝贵意见。
感谢在座各位的聆听。