信息学中的数学思维

demerzel

2018 年 8 月 12 日

简介

- 对式子中的一部分拿出来求和是一个常用策略。

简介

- 对式子中的一部分拿出来求和是一个常用策略。
- 比如说，假设现在要求 $\sum_{i=1}^{100} k \cdot i$，那么就可以先求 $\sum_{i=1}^{100} i$ 然后再乘上 k。

简介

- 对式子中的一部分拿出来求和是一个常用策略。
- 比如说，假设现在要求 $\sum_{i=1}^{100} k \cdot i$，那么就可以先求 $\sum_{i=1}^{100} i$ 然后再乘上 k。
- 所以这本质就是交换和号而已。

简介

- 对式子中的一部分拿出来求和是一个常用策略。
- 比如说，假设现在要求 $\sum_{i=1}^{100} k \cdot i$，那么就可以先求 $\sum_{i=1}^{100} i$ 然后再乘上 k。
- 所以这本质就是交换和号而已。
- 下面来看一道比较典型的例题。

来源：wearry 出的

定义一个无向图的权值为联通块个数的 k 次方，求所有 n 个点的无向图的权值和。

$n \leq 10^3, k \leq 20$

题解

- 首先设 g_n 表示 n 个点无向图的数量，这个是可以求的。

题解

- 首先设 g_n 表示 n 个点无向图的数量，这个是可以求的。
- 直观上可以设 $f[n][k]$ 表示 n 个点的无向图联通块的 k 次方和，那么转移时枚举 1 号点所在联通块大小：

$$
\begin{aligned}
f[n][k] &= \sum_{i=1}^{n} g_i \cdot \sum_{\text{所有 n-i 个点的图 G}} (c_G+1)^k \\
&= \sum_{i=1}^{n} g_i \cdot \sum_{\text{所有 n-i 个点的图 G}} \sum_{j=0}^{k} \binom{k}{j} c_G^j \\
&= \sum_{i=1}^{n} g_i \sum_{j=0}^{k} \binom{k}{j} \cdot f[n-i][j]
\end{aligned}
$$

题解

- 首先设 g_n 表示 n 个点无向图的数量，这个是可以求的。
- 直观上可以设 $f[n][k]$ 表示 n 个点的无向图联通块的 k 次方和，那么转移时枚举 1 号点所在联通块大小：

$$\begin{aligned}
f[n][k] &= \sum_{i=1}^{n} g_i \cdot \sum_{\text{所有 n-i 个点的图 G}} (c_G+1)^k \\
&= \sum_{i=1}^{n} g_i \cdot \sum_{\text{所有 n-i 个点的图 G}} \sum_{j=0}^{k} \binom{k}{j} c_G^j \\
&= \sum_{i=1}^{n} g_i \sum_{j=0}^{k} \binom{k}{j} \cdot f[n-i][j]
\end{aligned}$$

- 这样就是一个 n^2k^2 的算法，然而这太慢了。

题解

- 运用常见方法：$c^k = \sum_{i=0}^{k} \left\{ {k \atop j} \right\} \binom{c}{j} j!$

题解

- 运用常见方法：$c^k = \sum_{i=0}^{k} \left\{ {k \atop j} \right\} \binom{c}{j} j!$
- 那么答案等于 $\sum_{\text{所有 n 个点的图 G}} \sum_{i=0}^{k} \left\{ {k \atop i} \right\} \binom{c_G}{j} j!$

题解

- 运用常见方法：$c^k = \sum_{i=0}^{k} \left\{ {k \atop j} \right\} \binom{c}{j} j!$
- 那么答案等于 $\sum_{\text{所有 n 个点的图 G}} \sum_{i=0}^{k} \left\{ {k \atop i} \right\} \binom{c_G}{j} j!$
- 那么只要求出 $\sum_{\text{所有 n-i 个点的图 G}} \binom{c_G}{j}$ 就可以了

题解

- 那么将 $f[n][k]$ 的含义改为所有 n 个点的图的 $\binom{\text{联通块数}}{k}$ 之和。

题解

- 那么将 $f[n][k]$ 的含义改为所有 n 个点的图的 $\binom{\text{联通块数}}{k}$ 之和。
- 然后转移的复杂度就少了一个 k:

$$f[n][k]=\sum_{i=1}^{n}g_i\sum_{j=0}^{k}f[n-i][k]+f[n-i][k-1]$$

- 这是由于组合数具有优秀的性质 $C_n^m+C_n^{m-1}=C_{n+1}^m$。

题解

- 那么将 $f[n][k]$ 的含义改为所有 n 个点的图的 $\binom{\text{联通块数}}{k}$ 之和。
- 然后转移的复杂度就少了一个 k:

$$f[n][k] = \sum_{i=1}^{n} g_i \sum_{j=0}^{k} f[n-i][k] + f[n-i][k-1]$$

- 这是由于组合数具有优秀的性质 $C_n^m + C_n^{m-1} = C_{n+1}^m$。
- 实际上这道题的复杂度可以做到更好，稍后会提到。

来源：沃·兹基·楚德

有 n 个盒子，每秒钟随机选择一个盒子放入一个球，期望多少秒后，每个盒子中的球都达到 k 个。

$n \le 50, k \le 2000$

题解

我搬出当时写的题解。

简介

考虑式子或式子中一部分的组合意义。

有时可以得到一条新的思路。

AGC001E BBQ Hard

给定 n 个二元组 (A_i, B_i)，求

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\binom{A_i+A_j+B_i+B_j}{A_i+A_j}$$

$n \le 2*10^5, A_i \le 2000, B_i \le 2000$

题解

- 式子里的组合数可以看做网格图中的路径条数。

题解

- 式子里的组合数可以看做网格图中的路径条数。
- 即从 $(-A_j, -B_j)$ 到 (A_i, B_i) 的路径条数。

题解

- 式子里的组合数可以看做网格图中的路径条数。
- 即从 $(-A_j, -B_j)$ 到 (A_i, B_i) 的路径条数。
- 设所有 (A_i, B_i) 的集合为 S，所有 $(-A_i, -B_i)$ 的集合为 T，那么要求的就是从 T 中选一个点，再从 S 中选一个点，之间路径方案的和。

题解

- 式子里的组合数可以看做网格图中的路径条数。
- 即从 $(-A_j, -B_j)$ 到 (A_i, B_i) 的路径条数。
- 设所有 (A_i, B_i) 的集合为 S，所有 $(-A_i, -B_i)$ 的集合为 T，那么要求的就是从 T 中选一个点，再从 S 中选一个点，之间路径方案的和。
- 可以先对任一点 (x, y) 求出从 T 中选一个点到 (x, y) 路径方案的和，由于坐标范围很小，可以直接 O(平方)dp。

题解

- 式子里的组合数可以看做网格图中的路径条数。
- 即从 $(-A_j, -B_j)$ 到 (A_i, B_i) 的路径条数。
- 设所有 (A_i, B_i) 的集合为 S，所有 $(-A_i, -B_i)$ 的集合为 T，那么要求的就是从 T 中选一个点，再从 S 中选一个点，之间路径方案的和。
- 可以先对任一点 (x, y) 求出从 T 中选一个点到 (x, y) 路径方案的和，由于坐标范围很小，可以直接 O(平方)dp。
- 然后枚举 S 中的点统计答案就可以了。

还是这道题

定义一个无向图的权值为联通块个数的 k 次方，求所有 n 个点的无向图的权值和。

$n \leq 10^5, k \leq 20$

题解

我搬出当时的题解。

简介

差分思想十分常见。

来源未知

一个长度为 n 的 01 序列，初始全为 0，每秒会等概率随机一个区间，将区间中的元素异或上 1，问期望多少秒后序列变为全 1。

$n \leq 10^6$

题解

- 将问题转化成初始为全 1，结束为全 0，这看起来没有用。

题解

- 将问题转化成初始为全 1，结束为全 0，这看起来没有用。
- 假如原序列为 A，构造 A 的差分序列 B，即满足 A 是 B 的异或前缀和。注意 B 的长度为 $n+1$ 而不是 n。

题解

- 将问题转化成初始为全 1，结束为全 0，这看起来没有用。
- 假如原序列为 A，构造 A 的差分序列 B，即满足 A 是 B 的异或前缀和。注意 B 的长度为 $n+1$ 而不是 n。
- 那么 A 区间异或上 1，对应 B 选择两个不同的位置各自异或上 1。

题解

- 将问题转化成初始为全 1，结束为全 0，这看起来没有用。
- 假如原序列为 A，构造 A 的差分序列 B，即满足 A 是 B 的异或前缀和。注意 B 的长度为 $n+1$ 而不是 n。
- 那么 A 区间异或上 1，对应 B 选择两个不同的位置各自异或上 1。
- 然后问题变成了这样：初始序列为 $1, 0, 0, 0, \ldots, 0, 1$，每秒选两个不同位置异或上 1，期望多少秒后序列变成全 0。

题解

- 令 $m = n + 1$ 即为 B 的长度，设 f_i 代表当前序列中有 i 个 1，期望过多少秒序列变为全 0。

题解

- 令 $m = n + 1$ 即为 B 的长度，设 f_i 代表当前序列中有 i 个 1，期望过多少秒序列变为全 0。
- 那么边界条件为 $f_0 = 0$，转移方程为 $f_i = \frac{C_i^2 f_{i-2} + C_{m-i}^2 f_{i+2} + i(m-i) f_i}{C_n^2} + 1$

题解

- 令 $m=n+1$ 即为 B 的长度，设 f_i 代表当前序列中有 i 个 1，期望过多少秒序列变为全 0。
- 那么边界条件为 $f_0=0$，转移方程为 $f_i=\frac{C_i^2 f_{i-2}+C_{m-i}^2 f_{i+2}+i(m-i)f_i}{C_n^2}+1$
- 运用常用方法，可以将每个 f_i 表示为 $k\cdot f_{i+2}+b$ 的形式，边界条件为 $f_0=0\cdot f_2+0$。

题解

- 令 $m = n + 1$ 即为 B 的长度，设 f_i 代表当前序列中有 i 个 1，期望过多少秒序列变为全 0。
- 那么边界条件为 $f_0 = 0$，转移方程为 $f_i = \frac{C_i^2 f_{i-2} + C_{m-i}^2 f_{i+2} + i(m-i) f_i}{C_n^2} + 1$
- 运用常用方法，可以将每个 f_i 表示为 $k \cdot f_{i+2} + b$ 的形式，边界条件为 $f_0 = 0 \cdot f_2 + 0$。
- 然后一路推过去，最后在 f_m 或 f_{m-1} 处解方程（取决于 m 的奇偶）。

题解

- 令 $m=n+1$ 即为 B 的长度，设 f_i 代表当前序列中有 i 个 1，期望过多少秒序列变为全 0。
- 那么边界条件为 $f_0=0$，转移方程为 $f_i=\frac{C_i^2 f_{i-2}+C_{m-i}^2 f_{i+2}+i(m-i)f_i}{C_n^2}+1$
- 运用常用方法，可以将每个 f_i 表示为 $k\cdot f_{i+2}+b$ 的形式，边界条件为 $f_0=0\cdot f_2+0$。
- 然后一路推过去，最后在 f_m 或 f_{m-1} 处解方程（取决于 m 的奇偶）。
- 然后再一路反推回去就可以了，答案就是 f_2。

AGC018E Sightseeing Plan

这题已经被讲过很多次了，这次就不讲了，有兴趣的同学可以去了解。这题中的差分思想也很有意思。

简介

没有任何思维的套路，主要内容就是凑式子。

AGC005F Many Easy Problems

给你一棵 n 个点的树和一个数字 k，对于树上的一个点集 S，定义 $f(S)$ 等于将 S 中的点连接成一个连通块所需的最小边数。从 n 个点种选出 k 个点共有 $\binom{n}{k}$ 种方法，现在要求这 $\binom{n}{k}$ 种方法的 f 值之和。

由于上面这个问题过于简单，对于所有 $k \in [1, n]$ 都要求出答案。

题解

- 考虑一条边对答案的贡献，假设这条边一侧有 x 个点，那么贡献是 $\binom{n}{k} - \binom{x}{k} - \binom{n-x}{k}$。

题解

- 考虑一条边对答案的贡献，假设这条边一侧有 x 个点，那么贡献是 $\binom{n}{k}-\binom{x}{k}-\binom{n-x}{k}$。
- 那么可以搞一个数组 b，每次 b_n 加 1，b_x 和 b_{n-x} 减 1。

题解

- 考虑一条边对答案的贡献，假设这条边一侧有 x 个点，那么贡献是 $\binom{n}{k} - \binom{x}{k} - \binom{n-x}{k}$。
- 那么可以搞一个数组 b，每次 b_n 加 1，b_x 和 b_{n-x} 减 1。
- 然后 k 的答案就可以这样算：$\sum_{i=1}^{n} b_i \cdot \binom{i}{k}$。

题解

- 多个 k 如何做呢，将式子化一下：

$$
\begin{aligned}
&\sum_{i=1}^{n} b_i \cdot \binom{i}{k} \\
=&\sum_{i=1}^{n} \frac{b_i i!}{k!(i-k)!} \\
=&\frac{1}{k!}\sum_{i=1}^{n} (b_i i!) \cdot [-(k-i)]!
\end{aligned}
$$

题解

- 多个 k 如何做呢，将式子化一下：

$$
\begin{aligned}
&\sum_{i=1}^{n} b_i \cdot \binom{i}{k} \\
=&\sum_{i=1}^{n} \frac{b_i i!}{k!(i-k)!} \\
=&\frac{1}{k!}\sum_{i=1}^{n} (b_i i!) \cdot [-(k-i)]!
\end{aligned}
$$

- 可以看到，若设 $f_i = b_i i!$，$g_i = (-i)!$，那么将 f 和 g 卷起来就得到想求的。

题解

- 多个 k 如何做呢，将式子化一下：

$$
\begin{aligned}
&\sum_{i=1}^{n} b_i \cdot \binom{i}{k} \\
=&\sum_{i=1}^{n} \frac{b_i i!}{k!(i-k)!} \\
=&\frac{1}{k!}\sum_{i=1}^{n}(b_i i!) \cdot [-(k-i)]!
\end{aligned}
$$

- 可以看到，若设 $f_i = b_i i!$，$g_i = (-i)!$，那么将 f 和 g 卷起来就得到想求的。
- 所以就可以 NTT，需要处理一下负数幂次。

简介

这是以前的比赛中见到的，不知道适用面广不广。

来源未知

有一长度为 n 的序列，每个元素在 $[1, m]$ 之间均匀随机。
设 a_i 为元素 i 出现的次数，求 $\prod_{i=1}^{m} a_i^k$ 的期望。
$m \leq n \leq 10^5$，$k \leq 10^5$
有 $k = 1$ 的部分分，可以从这里入手。

题解 k=1

- 设 $c_{i,j}$ 表示第 i 个元素是否为 j，$c_{i,j}$ 只有 0 和 1 两种取值，且 $c_{i,j}$ 和 $c_{i,k}$（$j \neq k$）不会同时为 1。

题解 k=1

- 设 $c_{i,j}$ 表示第 i 个元素是否为 j，$c_{i,j}$ 只有 0 和 1 两种取值，且 $c_{i,j}$ 和 $c_{i,k}$（$j \neq k$）不会同时为 1。
- 那么 $a_i = \sum_{j=1}^{n} c_{j,i}$，所以 $\prod_{i=1}^{m} a_i = \prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})$。

题解 k=1

- 设 $c_{i,j}$ 表示第 i 个元素是否为 j，$c_{i,j}$ 只有 0 和 1 两种取值，且 $c_{i,j}$ 和 $c_{i,k}$（$j \neq k$）不会同时为 1。
- 那么 $a_i = \sum_{j=1}^{n} c_{j,i}$，所以 $\prod_{i=1}^{m} a_i = \prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})$。
- 可以将乘积展开成单项式，然后分别求每一项的期望：$\sum_{all\,\{b_m\}} \prod_{i=1}^{m} c_{b_i,i}$

题解 k=1

- 设 $c_{i,j}$ 表示第 i 个元素是否为 j，$c_{i,j}$ 只有 0 和 1 两种取值，且 $c_{i,j}$ 和 $c_{i,k}$（$j \neq k$）不会同时为 1。
- 那么 $a_i = \sum_{j=1}^{n} c_{j,i}$，所以 $\prod_{i=1}^{m} a_i = \prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})$。
- 可以将乘积展开成单项式，然后分别求每一项的期望：$\sum_{all\,\{b_m\}} \prod_{i=1}^{m} c_{b_i,i}$
- 注意到如果 $\{b_m\}$ 中有两个相同元素，则那一项的取值恒为 0，那么只需要考虑互不相同的情况。

题解 k=1

- 设 $c_{i,j}$ 表示第 i 个元素是否为 j，$c_{i,j}$ 只有 0 和 1 两种取值，且 $c_{i,j}$ 和 $c_{i,k}$（$j \neq k$）不会同时为 1。
- 那么 $a_i = \sum_{j=1}^{n} c_{j,i}$，所以 $\prod_{i=1}^{m} a_i = \prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})$。
- 可以将乘积展开成单项式，然后分别求每一项的期望：$\sum_{all\,\{b_m\}} \prod_{i=1}^{m} c_{b_i,i}$
- 注意到如果 $\{b_m\}$ 中有两个相同元素，则那一项的取值恒为 0，那么只需要考虑互不相同的情况。
- 若 $a \neq b$，那么 $c_{a,i}$ 与 $c_{b,j}$ 是互相独立的事件，所以 $E(\prod_{i=1}^{m} c_{b_i,i}) = \prod_{i=1}^{m} E(c_{b_i,i})$

题解 k=1

- 设 $c_{i,j}$ 表示第 i 个元素是否为 j，$c_{i,j}$ 只有 0 和 1 两种取值，且 $c_{i,j}$ 和 $c_{i,k}$（$j \neq k$）不会同时为 1。
- 那么 $a_i = \sum_{j=1}^{n} c_{j,i}$，所以 $\prod_{i=1}^{m} a_i = \prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})$。
- 可以将乘积展开成单项式，然后分别求每一项的期望：$\sum_{all\,\{b_m\}} \prod_{i=1}^{m} c_{b_i,i}$
- 注意到如果 $\{b_m\}$ 中有两个相同元素，则那一项的取值恒为 0，那么只需要考虑互不相同的情况。
- 若 $a \neq b$，那么 $c_{a,i}$ 与 $c_{b,j}$ 是互相独立的事件，所以 $E(\prod_{i=1}^{m} c_{b_i,i}) = \prod_{i=1}^{m} E(c_{b_i,i})$
- 显然任意 $E(c_{i,j}) = \frac{1}{m}$，所以单项的期望为 $(\frac{1}{m})^m$，因为这样的项共有 $n^{\underline{m}}$ 项，所以答案就是 $n^{\underline{m}} \cdot (\frac{1}{m})^m$。

题解

- 现在变成了这样

$$\prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})^k$$

$$= \sum_{all\,\{b\}} \prod_{i=1}^{m}\prod_{j=1}^{k} c_{b_{i,j},i}$$

题解

- 现在变成了这样

$$\prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})^k$$

$$= \sum_{all\,\{b\}} \prod_{i=1}^{m} \prod_{j=1}^{k} c_{b_{i,j},i}$$

- 虽然 $b_{a,j}$ 和 $b_{b,j}$（$a \neq b$）不能相等，但是 $b_{a,i}$ 和 $b_{a,j}$ 是可以相等的，并且每一项的贡献变成了 $(\frac{1}{m})^{b\text{ 中不同元素个数}}$。

题解

- 现在变成了这样

$$\prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})^k$$

$$= \sum_{all\,\{b\}} \prod_{i=1}^{m}\prod_{j=1}^{k} c_{b_{i,j},i}$$

- 虽然 $b_{a,j}$ 和 $b_{b,j}$（$a \neq b$）不能相等，但是 $b_{a,i}$ 和 $b_{a,j}$ 是可以相等的，并且每一项的贡献变成了 $(\frac{1}{m})^{b\text{ 中不同元素个数}}$。
- 于是我们要对于每一个 r 统计 b 中不同元素个数为 r 的方案数。

题解

- 现在变成了这样

$$\prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})^k$$

$$= \sum_{all\{b\}} \prod_{i=1}^{m}\prod_{j=1}^{k} c_{b_{i,j},i}$$

- 虽然 $b_{a,j}$ 和 $b_{b,j}$（$a \neq b$）不能相等，但是 $b_{a,i}$ 和 $b_{a,j}$ 是可以相等的，并且每一项的贡献变成了 $(\frac{1}{m})^{b\text{ 中不同元素个数}}$。
- 于是我们要对于每一个 r 统计 b 中不同元素个数为 r 的方案数。
- 先不考虑具体的值，设 $f[i][j]$ 表示将前 $b_{i,*}$ 划为 j 个不同集合的方案数，那么转移就是 $f[i][j] \cdot \left\{{k \atop l}\right\} \to f[i+1][j+l]$。最后 $n^{\underline{r}} \cdot f[m][r]$ 就是方案数。

题解

- 现在变成了这样

$$\prod_{i=1}^{m}(\sum_{j=1}^{n} c_{j,i})^k$$

$$= \sum_{all\,\{b\}} \prod_{i=1}^{m} \prod_{j=1}^{k} c_{b_{i,j},i}$$

- 虽然 $b_{a,j}$ 和 $b_{b,j}$（$a \neq b$）不能相等，但是 $b_{a,i}$ 和 $b_{a,j}$ 是可以相等的，并且每一项的贡献变成了 $(\frac{1}{m})^{\text{b 中不同元素个数}}$。
- 于是我们要对于每一个 r 统计 b 中不同元素个数为 r 的方案数。
- 先不考虑具体的值，设 $f[i][j]$ 表示将前 $b_{i,*}$ 划为 j 个不同集合的方案数，那么转移就是 $f[i][j] \cdot \left\{ {k \atop l} \right\} \to f[i+1][j+l]$。最后 $n^{\underline{r}} \cdot f[m][r]$ 就是方案数。
- 上面这个当然可以拿 NTT 优化一下，就可以了。