Problem 4: Counting

张天昀

南京大学计算机科学与技术系
171860508@smail.nju.edu.cn

2018 年 9 月 3 日

考虑整数 $1 \dots n$ 的排列 $a_1 a_2 \dots a_n$.

定义（逆序对）

如果 $i < j$ 且 $a_i > a_j$，则称 (i, j) 是该排列的一个逆序对。

定义（逆序表）

记 $b_j = |\{a_i | i < j \wedge a_i > a_j\}|$，则称 $b_1 b_2 \dots b_n$ 为该排列的逆序表。

(1) 证明：在“$1 \dots n$ 的所有排列”与“大小为 n 的所有逆序表”之间存在双射。

(2) 记 $I_n(k)$ 为恰有 k 个逆序对的 $1 \dots n$ 的排列的数目。求 $I_n(0)$、$I_n(1)$、$I_n(2)$ 和 $I_n(3)$。

Phase 1

首先回忆如何证明双射。

Definition

定义一个映射 $f\colon \{\pi\} \to \{\sigma\}$，
其中 $f((a_1, a_2, \ldots, a_n)) = (b_1, b_2, \ldots, b_n)$.
即对排列 π，$f(\pi) = \pi$ 对应的逆序表。

Phase 1

首先回忆如何证明双射。

Definition

定义一个映射 $f\colon \{\pi\} \to \{\sigma\}$,
其中 $f((a_1, a_2, \ldots, a_n)) = (b_1, b_2, \ldots, b_n)$.
即对排列 π，$f(\pi) = \pi$ 对应的逆序表。

要证明 f 是双射，只要证明 f 既是单射又是满射。

Injective

Prove that

$$\forall \pi_1, \pi_2 \in \{\pi\}, f(\pi_1) = f(\pi_2) \Rightarrow \pi_1 = \pi_2$$

Surjective

Prove that

$$\forall \sigma \in \{\sigma\}, \exists \pi_0 \in \{\pi\} \wedge f(\pi_0) = \sigma$$

Injective

由 $f(\pi_1) = f(\pi_2)$，有两个完全相等的逆序表

$$(b_{1_1}, b_{1_2}, \ldots, b_{1_n}) = (b_{2_1}, b_{2_2}, \ldots, b_{2_n})$$

Injective

由 $f(\pi_1) = f(\pi_2)$，有两个完全相等的逆序表

$$(b_{1_1}, b_{1_2}, \ldots, b_{1_n}) = (b_{2_1}, b_{2_2}, \ldots, b_{2_n})$$

先考虑最后一个数，由逆序表的定义得到

$$b_{1_n} = |\{a_{1_i} | i < n \wedge a_{1_i} > a_{1_n}\}|$$

可知 $a_{1_1}, a_{1_2}, \ldots, a_{1_{n-1}}$ 中有 b_{1_n} 个数满足 $a_{1_i} > a_{1_n}$。
所以 a_{1_n} 是 $S_n = \{1, 2, \ldots, n\}$ 中第 $b_{1_n} + 1$ 大的数，即

$$a_{1_n} = n - b_{1_n}$$

Injective

由 $f(\pi_1) = f(\pi_2)$，有两个完全相等的逆序表

$$(b_{1_1}, b_{1_2}, \ldots, b_{1_n}) = (b_{2_1}, b_{2_2}, \ldots, b_{2_n})$$

先考虑最后一个数，由逆序表的定义得到

$$b_{1_n} = |\{a_{1_i} | i < n \wedge a_{1_i} > a_{1_n}\}|$$

可知 $a_{1_1}, a_{1_2}, \ldots, a_{1_{n-1}}$ 中有 b_{1_n} 个数满足 $a_{1_i} > a_{1_n}$。
所以 a_{1_n} 是 $S_n = \{1, 2, \ldots, n\}$ 中第 $b_{1_n} + 1$ 大的数，即

$$a_{1_n} = n - b_{1_n}$$

同理，有

$$a_{2_n} = n - b_{2_n}$$

所以

$$b_{1_n} = b_{2_n} \Rightarrow a_{1_n} = a_{2_n}$$

Injective

考虑第 $n-1$ 个数，定义

$$S_{n-1} = S_n \setminus \{a_{1_n}\}$$

同理可得

$$a_{1_{n-1}} = (b_{1_{n-1}} + 1)\text{-th of } S_{n-1}$$

$$a_{2_{n-1}} = (b_{2_{n-1}} + 1)\text{-th of } S_{n-1}$$

所以

$$b_{1_{n-1}} = b_{2_{n-1}} \Rightarrow a_{1_{n-1}} = a_{2_{n-1}}$$

Injective

考虑第 $n-1$ 个数，定义

$$S_{n-1} = S_n \setminus \{a_{1_n}\}$$

同理可得

$$a_{1_{n-1}} = (b_{1_{n-1}} + 1)\text{-th of } S_{n-1}$$

$$a_{2_{n-1}} = (b_{2_{n-1}} + 1)\text{-th of } S_{n-1}$$

所以

$$b_{1_{n-1}} = b_{2_{n-1}} \Rightarrow a_{1_{n-1}} = a_{2_{n-1}}$$

递推可得

$$\forall i \in \mathbb{N} \cap [1, n], a_{1_i} = a_{2_i}$$

即

$$\pi_1 = (a_{1_1}, a_{1_2}, \ldots, a_{1_n}) = (a_{2_1}, a_{2_2}, \ldots, a_{2_n}) = \pi_2$$

Surjective

由刚才的证明过程知，对 $\forall\sigma = (b_1, b_2, \ldots, b_n) \in \{\sigma\}$，可以构造如下排列 $\pi_0 = (a_1, a_2, \ldots, a_n)$ 满足 $f(\pi_0) = \sigma$：

$$a_i = (b_i + 1)\text{-th of } S_i$$

$$S_i = \{1, 2, \ldots, n\} \setminus \bigcup_{k=i+1}^{n} \{a_k\}$$

Surjective

由刚才的证明过程知，对 $\forall\sigma = (b_1, b_2, \ldots, b_n) \in \{\sigma\}$，可以构造如下排列 $\pi_0 = (a_1, a_2, \ldots, a_n)$ 满足 $f(\pi_0) = \sigma$：

$$a_i = (b_i + 1)\text{-th of } S_i$$

$$S_i = \{1, 2, \ldots, n\} \setminus \bigcup_{k=i+1}^{n} \{a_k\}$$

所有 $1 \sim n$ 的数字在 π_0 中均出现且仅出现了一次，所以 $\pi_0 \in \{\pi\}$。 $\square$

Phase 2 - Method 0

想法 0（暴力）

枚举 $n!$ 个排列数数。

Phase 2 - Method 0

想法 0（暴力）

枚举 $n!$ 个排列数数。

复杂度最快也要 $O(n!n\lg n)$，而且得不到公式。

Phase 2 - Method 1

想法 1（直观）

逆序对的产生一定是交换了排列中的两个元素的位置。每交换一对递增的相邻元素就会产生一组逆序对，所以通过枚举所有的交换的方法就可以得到排列的个数。

Phase 2 - Method 1

想法 1（直观）

逆序对的产生一定是交换了排列中的两个元素的位置。每交换一对递增的相邻元素就会产生一组逆序对，所以通过枚举所有的交换的方法就可以得到排列的个数。

0) $k = 0$ 表示整个排列没有逆序对，所以排列是有序的，所以

$$I_n(0) = 1$$

想法 1（直观）

逆序对的产生一定是交换了排列中的两个元素的位置。每交换一对递增的相邻元素就会产生一组逆序对，所以通过枚举所有的交换的方法就可以得到排列的个数。

0) $k = 0$ 表示整个排列没有逆序对，所以排列是有序的，所以

$$I_n(0) = 1$$

1) 通过交换排列中的相邻元素可以产生 1 组逆序对，共有 $\binom{n-1}{1} = n - 1$ 种选法，所以

$$I_n(1) = \binom{n-1}{1} = n - 1$$

Phase 2 - Method 1

2) 产生 2 组逆序对需要交换两组相邻的元素。

Phase 2 - Method 1

2) 产生 2 组逆序对需要交换两组相邻的元素。
如果两组元素不相交，每种选法只包含 1 个排列；

2) 产生 2 组逆序对需要交换两组相邻的元素。
如果两组元素不相交，每种选法只包含 1 个排列；
如果两组元素相交，那么交换顺序是会对结果造成影响的，如

$$1\ 2\ 3 \rightarrow \begin{cases} \underline{2\ 1}\ 3 \rightarrow 2\ \underline{3\ 1} \\ 1\ \underline{3\ 2} \rightarrow \underline{3\ 1}\ 2 \end{cases}$$

所以每种选法会包含 2 个排列。

2) 产生 2 组逆序对需要交换两组相邻的元素。
如果两组元素不相交，每种选法只包含 1 个排列；
如果两组元素相交，那么交换顺序是会对结果造成影响的，如

$$1\ 2\ 3 \to \begin{cases} \underline{2\ 1}\ 3 \to 2\ \underline{3\ 1} \\ 1\ \underline{3\ 2} \to \underline{3\ 1}\ 2 \end{cases}$$

所以每种选法会包含 2 个排列。
利用**隔板法**可以算出正确的结果（把剩下的数放入区间中）。

$$\begin{aligned} I_n(2) &= A + 2B \\ &= \binom{n-4+3-1}{2} + 2 \times \binom{n-3+2-1}{1} \\ &= \frac{n^2-5n+6}{2} + 2(n-2) = \frac{n^2-n-2}{2} \end{aligned}$$

Phase 2 - Method 1

3) $k = 3$ 时，首先考虑互不相交的方法数 A，每种选法包含 1 个排列；
出现一次相交的（2+3）方法数 B，每种选法包含 2 个排列；
出现两次相交的（3 连）方法数 C，每种选法包含 1 个排列；
出现两次相交的（4 连）方法数 D，每种选法包含 4 个排列；

Phase 2 - Method 1

3) $k=3$ 时，首先考虑互不相交的方法数 A，每种选法包含 1 个排列；
出现一次相交的（2+3）方法数 B，每种选法包含 2 个排列；
出现两次相交的（3 连）方法数 C，每种选法包含 1 个排列；
出现两次相交的（4 连）方法数 D，每种选法包含 4 个排列；

为什么穷举出来是 6 个而这里只包含 4 个？

$$1\ 2\ 3\ 4 \rightarrow \begin{cases} 1\ 4\ 3\ 2 \\ 2\ 3\ 4\ 1 \\ 2\ 4\ 1\ 3 \\ 3\ 1\ 4\ 2 \\ 3\ 2\ 1\ 4 \\ 4\ 1\ 2\ 3 \end{cases}$$

Phase 2 - Method 1

3) $k=3$ 时，首先考虑互不相交的方法数 A，每种选法包含 1 个排列；
出现一次相交的（2+3）方法数 B，每种选法包含 2 个排列；
出现两次相交的（3 连）方法数 C，每种选法包含 1 个排列；
出现两次相交的（4 连）方法数 D，每种选法包含 4 个排列；

为什么穷举出来是 6 个而这里只包含 4 个?

$$1\ 2\ 3\ 4 \rightarrow \begin{cases} 1\ 4\ 3\ 2 \\ 2\ 3\ 4\ 1 \\ 2\ 4\ 1\ 3 \\ 3\ 1\ 4\ 2 \\ 3\ 2\ 1\ 4 \\ 4\ 1\ 2\ 3 \end{cases}$$

有两个与 C 中的重复了

3) 注意算 B 的时候两次选择是有先后顺序的

$$
\begin{aligned}
I_n(3) &= A + 2B + C + 4D \\
&= \binom{n-6+4-1}{3} + 2 \times 2! \times \binom{n-5+3-1}{2} \\
&\quad + \binom{n-3+2-1}{1} + 4 \times \binom{n-4+2-1}{1} \\
&= \frac{n^3 - 12n^2 + 47n - 60}{6} + (2n^2 - 9n + 10) \\
&= \frac{n^3 - 7n}{6}
\end{aligned}
$$

3) 注意算 B 的时候两次选择是有先后顺序的

$$
\begin{aligned}
I_n(3) &= A + 2B + C + 4D \\
&= \binom{n-6+4-1}{3} + 2 \times 2! \times \binom{n-5+3-1}{2} \\
&\quad + \binom{n-3+2-1}{1} + 4 \times \binom{n-4+2-1}{1} \\
&= \frac{n^3 - 12n^2 + 47n - 60}{6} + (2n^2 - 9n + 10) \\
&= \frac{n^3 - 7n}{6}
\end{aligned}
$$

评价：组合数坑实在太多了，根本没法一次算对。

Phase 2 - Method 2

既然有第一问，那么第一问和第二问之间肯定有什么联系。

Phase 2 - Method 2

既然有第一问，那么第一问和第二问之间肯定有什么联系。

想法 2（逆序表）

注意到逆序对个数 $k = \sum_{i=1}^{n} b_i$，排列 π 和逆序表 σ 又存在一一对应的关系，那么只要枚举所有满足条件的逆序表 σ 就可以了。

Phase 2 - Method 2

既然有第一问，那么第一问和第二问之间肯定有什么联系。

想法 2（逆序表）

注意到逆序对个数 $k = \sum_{i=1}^{n} b_i$，排列 π 和逆序表 σ 又存在一一对应的关系，那么只要枚举所有满足条件的逆序表 σ 就可以了。

i	1	2	3	…
π	n	$n-1$	$n-2$	…
σ	0	1	2	…

又注意到 $b_i < i$（因为 a_i 前面最多只可能有 $i-1$ 个数比他大），所以很好枚举。

Phase 2 - Method 2

0) $k = 0$，所有的 $b_i = 0$，所以

$$I_n(0) = 1$$

Phase 2 - Method 2

0) $k = 0$，所有的 $b_i = 0$，所以

$$I_n(0) = 1$$

1) $k = 1$，说明有一个 $b_j = 1$，其余 $b_i = 0$。
由于 $b_1 < 1 \Rightarrow j \neq 1 \Rightarrow j \in \{2, \ldots, n\}$，所以

$$I_n(1) = \binom{n-1}{1} = n - 1$$

Phase 2 - Method 2

0) $k = 0$，所有的 $b_i = 0$，所以

$$I_n(0) = 1$$

1) $k = 1$，说明有一个 $b_j = 1$，其余 $b_i = 0$。
由于 $b_1 < 1 \Rightarrow j \neq 1 \Rightarrow j \in \{2, \ldots, n\}$，所以

$$I_n(1) = \binom{n-1}{1} = n - 1$$

2) $k = 2$，可能 $b_{j_1} = b_{j_2} = 1$，其余 $b_i = 0$，或 $b_k = 2$，其余 $b_i = 0$。
由于 $b_1 < 1, b_2 < 2 \Rightarrow j_1, j_2 \in \{2, \ldots, n\}, k \in \{3, \ldots, n\}$ 所以

$$I_n(2) = \binom{n-1}{2} + \binom{n-2}{1} = \frac{n^2 - n - 2}{2}$$

Phase 2 - Method 2

0) $k = 0$，所有的 $b_i = 0$，所以

$$I_n(0) = 1$$

1) $k = 1$，说明有一个 $b_j = 1$，其余 $b_i = 0$。
由于 $b_1 < 1 \Rightarrow j \neq 1 \Rightarrow j \in \{2, \ldots, n\}$，所以

$$I_n(1) = \binom{n-1}{1} = n - 1$$

2) $k = 2$，可能 $b_{j_1} = b_{j_2} = 1$，其余 $b_i = 0$，或 $b_k = 2$，其余 $b_i = 0$。
由于 $b_1 < 1, b_2 < 2 \Rightarrow j_1, j_2 \in \{2, \ldots, n\}, k \in \{3, \ldots, n\}$ 所以

$$I_n(2) = \binom{n-1}{2} + \binom{n-2}{1} = \frac{n^2 - n - 2}{2}$$

3) $k = 3 = 1 + 1 + 1 = 1 + 2 = 3$，所以

$$I_n(3) = \binom{n-1}{3} + \binom{n-2}{1}\binom{n-2}{1} + \binom{n-3}{1} = \frac{n^3 - 7n}{6}$$

根本没用到的神秘的提示 $\sum_{0\leq k\leq n}\binom{k}{m}=\binom{n+1}{m+1}$ 到底有什么用呢?

Phase 2 - Method 3

根本没用到的神秘的提示 $\sum_{0\le k\le n}\binom{k}{m}=\binom{n+1}{m+1}$ 到底有什么用呢?

上网一搜，竟然发现一道 OJ 题。

LeetCode 629

Given two integers n and k, find how many different arrays consist of numbers from 1 to n such that there are exactly k inverse pairs.
The integer n is in the range $[1, 1000]$ and k is in the range $[0, 1000]$.

根本没用到的神秘的提示 $\sum_{0\le k\le n}\binom{k}{m}=\binom{n+1}{m+1}$ 到底有什么用呢?

上网一搜，竟然发现一道 OJ 题。

LeetCode 629

Given two integers n and k, find how many different arrays consist of numbers from 1 to n such that there are exactly k inverse pairs.
The integer n is in the range $[1, 1000]$ and k is in the range $[0, 1000]$.

10^3 的数据范围，大概就是 $O(nk)$ 或者类似 $O(nk\lg n)$ 的复杂度。

Phase 2 - Method 3

想法 3（动态规划）

$$ans[n][k] = \sum_{i=max\{0,k-n+1\}}^{k} ans[n-1][i]$$

想法 3（动态规划）

$$ans[n][k] = \sum_{i=max\{0,k-n+1\}}^{k} ans[n-1][i]$$

Phase 2 - Method 3

为什么答案可以递归求出呢?

(n,k) 的排列可以通过把 n 插入到 $(n-1,k)$ 的末尾得到;
也可以通过把 n 插入到 $(n-1,k-1)$ 的排列的倒数第二位得到;
也可以通过把 n 插入到 $(n-1,k-2)$ 的排列的倒数第三位得到……

Phase 2 - Method 3

为什么答案可以递归求出呢?

(n, k) 的排列可以通过把 n 插入到 $(n-1, k)$ 的末尾得到;
也可以通过把 n 插入到 $(n-1, k-1)$ 的排列的倒数第二位得到;
也可以通过把 n 插入到 $(n-1, k-2)$ 的排列的倒数第三位得到$\cdots\cdots$

但是不能通过将 n 插入到 $(n-1, k-n)$ 的排列中的得到，因为最多只会增加 $n-1$ 组逆序对。

为什么答案可以递归求出呢?

(n, k) 的排列可以通过把 n 插入到 $(n-1, k)$ 的末尾得到;
也可以通过把 n 插入到 $(n-1, k-1)$ 的排列的倒数第二位得到;
也可以通过把 n 插入到 $(n-1, k-2)$ 的排列的倒数第三位得到$\cdots\cdots$

但是不能通过将 n 插入到 $(n-1, k-n)$ 的排列中的得到，因为最多只会增加 $n-1$ 组逆序对。

所以递推式 $ans[n][k] = \sum_{i=max\{0,k-n+1\}}^{k} ans[n-1][i]$ 成立。

而这个计算过程中有大量的重复计算，适合记忆化搜索/DP。

Phase 2 - Method 3

初始条件 $I_n(0) = 1, I_0(k) = 0$。

初始条件 $I_n(0) = 1, I_0(k) = 0$。

0) 由上可知

$$I_n(0) = 1$$

Phase 2 - Method 3

初始条件 $I_n(0)=1, I_0(k)=0$。

0) 由上可知

$$I_n(0)=1$$

1) $I_n(1)=I_{n-1}(0)+I_{n-1}(1)=I_{n-1}(1)+1$，**注意首项是** $I_2(1)=1$
求解等差数列可得

$$I_n(1)=n-1$$

Phase 2 - Method 3

初始条件 $I_n(0) = 1, I_0(k) = 0$。

0) 由上可知

$$I_n(0) = 1$$

1) $I_n(1) = I_{n-1}(0) + I_{n-1}(1) = I_{n-1}(1) + 1$，**注意首项是** $I_2(1) = 1$
求解等差数列可得

$$I_n(1) = n - 1$$

2) $I_n(2) = I_{n-1}(0) + I_{n-1}(1) + I_{n-1}(2) = I_{n-1}(2) + n - 1$，$I_3(2) = 2$
求解二阶等差数列可得

$$I_n(2) = \frac{n^2 - n - 2}{2}$$

3) $I_n(3) = I_{n-1}(0) + I_{n-1}(1) + I_{n-1}(2) + I_{n-1}(3) = I_{n-1}(3) + \frac{n^2-n-2}{2}$，首项 $I_3(3) = 1$，这里可以利用提示求解三阶等差数列，得

$$
\begin{aligned}
I_n(3) &= \sum_{i=4}^{n} I_i(2) + I_3(3) \\
&= \sum_{i=4}^{n} \left(\binom{i-1}{2} + \binom{i-2}{1} \right) + 1 \\
&= \binom{n}{3} - \binom{3}{3} + \binom{n-1}{2} - \binom{2}{2} + 1 \\
&= \frac{n^3 - 3n^2 + 2n}{6} + \frac{n^2 - 3n + 2}{2} - 1 = \frac{n^3 - 7n}{6}
\end{aligned}
$$

3) $I_n(3) = I_{n-1}(0) + I_{n-1}(1) + I_{n-1}(2) + I_{n-1}(3) = I_{n-1}(3) + \frac{n^2-n-2}{2}$,
首项 $I_3(3) = 1$，这里可以利用提示求解三阶等差数列，得

$$\begin{aligned} I_n(3) &= \sum_{i=4}^{n} I_i(2) + I_3(3) \\ &= \sum_{i=4}^{n} \left(\binom{i-1}{2} + \binom{i-2}{1} \right) + 1 \\ &= \binom{n}{3} - \binom{3}{3} + \binom{n-1}{2} - \binom{2}{2} + 1 \\ &= \frac{n^3 - 3n^2 + 2n}{6} + \frac{n^2 - 3n + 2}{2} - 1 = \frac{n^3 - 7n}{6} \end{aligned}$$

当然，不用提示求解这个等差数列也不难⋯⋯

Phase 2 - Method 3 - Code

但其实这个算法交给机器跑是 $O(nk^2)$ 的，还是会 TLE。

再观察一下就会发现递推式做了太多次求和，用前缀和把时间复杂度砍到 $O(nk)$ 就跑得非常快了。

Phase 2 - Method 3 - Code

但其实这个算法交给机器跑是 $O(nk^2)$ 的，还是会 TLE。

再观察一下就会发现递推式做了太多次求和，用前缀和把时间复杂度砍到 $O(nk)$ 就跑得非常快了。

写得丑跑得慢的代码

```
class Solution {
public:
  int kInversePairs(int n, int k) {
    long long modulo = (long long) 1e9+7;
    long long ans[1005][1005] = {};
    for(int i = 1; i <= n; i++) {
      ans[i][1] = 1;
      for(int j = 1; j <= min(k, i * (i - 1) / 2); j++) {
        ans[i][j + 1] = ans[i][j];
        ans[i][j + 1] += ans[i - 1][j + 1]  - ans[i - 1][max(0, j - i + 1)];
        ans[i][j + 1] = (ans[i][j + 1] + modulo) % modulo;
      }
      for(int j = min(k, i * (i - 1) / 2) + 1; j <= min(k, 1000); j++) {
        ans[i][j + 1] = ans[i][j];
      }
    }
    return (ans[n][k + 1] - ans[n][k] + modulo) % modulo;
  }
};
```