# lydsy Monthly, November 2017

解题报告

2017 年 12 月 7 日

## 1 组题

设 $s_i$ 为 $a$ 的前缀和，枚举每个位置 $i$ 作为右端点，则左端点 $j$ 需要满足 $0 \le j \le i-k$ 且 $\frac{s_i-s_j}{i-j}$ 最大。

显然最优的 $j$ 只在凸壳上取到，用单调栈维护凸壳，每次查询的时候在凸壳上二分即可。

时间复杂度 $O(n \log n)$。

## 2 摘苹果

根据题意，第一步走到点 $i$ 的概率为 $\frac{d_i}{2m}$。考虑第一步位于 $i$，第二步从 $i$ 走到 $j$ 的概率，为 $\frac{d_i}{2md_i} = \frac{1}{2m}$。一共有 $d_j$ 个 $i$ 可以走到 $j$，故第二步位于点 $j$ 的概率为 $\frac{d_j}{2m}$。因此无论走多少步，位于点 $i$ 的概率恒为 $\frac{d_i}{2m}$。

$$ans = k\sum_{i=1}^{n}\frac{a_i d_i}{2m}$$

时间复杂度 $O(n+m)$。

## 3 分割序列

设 $s_i$ 为 $b$ 的前缀异或和，则对于前缀 $i$，要找一个 $j(0 \le j \le i)$ 使得 $s_j + (s_i \oplus s_j)$ 最大。

考虑二进制下从高位到低位贪心。若 $s_i$ 当前位是 0，那么 $s_j$ 这一位取 1 最优，否则取 0 取 1 都没有关系。问题转化为判定是否存在可行的 $j$，满足指定的若干位必须是 1。设 $f[S]$ 表示 $S$ 集合的位必须是 1 的 $j$ 的最小值，可以通过递推预处理求出。那么只要 $f[S] \le i$ 则可以取 1。

时间复杂度 $O((n+a)\log a)$。

## 4 图的价值

显然每个点的贡献都相等，枚举 1 号点的度数 $d$，则要选 $C(n-1,d)$ 个点与它连边，剩下 $n-1$ 个点之间随意连边，故：

$$ans = n2^{\frac{(n-1)(n-2)}{2}} \sum_{i=0}^{n-1} C(n-1,i)i^k$$

根据：

$$x^k = \sum_{i=1}^{k} Stirling2(k,i)i!C(x,i)$$

则

$$\begin{aligned} ans &= n2^{\frac{(n-1)(n-2)}{2}} \sum_{i=0}^{n-1} C(n-1,i)i^k \\ &= n2^{\frac{(n-1)(n-2)}{2}} \sum_{i=0}^{n-1} C(n-1,i) \sum_{j=1}^{k} Stirling2(k,j)j!C(i,j) \\ &= n2^{\frac{(n-1)(n-2)}{2}} \sum_{j=1}^{k} Stirling2(k,j)j! \sum_{i=0}^{n-1} C(n-1,i)C(i,j) \end{aligned}$$

考虑 $\sum C(n-1,i)C(i,j)$ 式子的组合意义：从 $n-1$ 个人中选择 $i$ 个人，再从中选择 $j$ 个人的方案数。这等价于，从 $n-1$ 个人中先选择 $j$ 个人，剩下 $n-1-j$ 个人可选可不选。故：

$$\sum_{i=0}^{n-1} \sum_{j=1}^{k} C(n-1,i)C(i,j) = \sum_{j=1}^{k} C(n-1,j)2^{n-1-j}$$

根据第二类 Stirling 数的卷积形式：

$$S(n,m) = \frac{1}{m!} \sum_{k=0}^{m} (-1)^k C(m,k)(m-k)^n = \sum_{k=0}^{m} \frac{(-1)^k (m-k)^n}{k!(m-k)!} = \sum_{k=0}^{m} \frac{(-1)^k}{k!} \times \frac{(m-k)^n}{(m-k)!}$$

可以通过 NTT 在 $O(k \log k)$ 的时间内预处理出所有 $Stirling2(k,j)$，而组合数则可以递推求出。

总时间复杂度 $O(k \log k)$。

## 5 硬盘检测

对于每种 $n$，在本地随机生成 100 组数据，计算不同元素个数的平均值，则与输入差值最小的那一个即为答案。

## 6 座位安排

若 $m$ 为奇数，则直接状压 DP，设 $f[i][j][k][l]$ 表示考虑前 $i$ 个座位，还需要放下 $j$ 集合的人，第 1 个座位是否坐人为 $k$，第 $l$ 个座位是否坐人为 $l$ 的方案数。

若 $m$ 为偶数，令 $h = \frac{m}{2}$，设 $f[i][j][k][l]$ 表示考虑前 $i$ 对座位，还需要放下 $j$ 集合的人，1 和 $1+h$ 坐人情况为 $k$，$i$ 和 $i+h$ 坐人情况为 $l$ 的方案数。

时间复杂度 $O(nm2^n)$。

## 7 删数游戏

因为数字互不相同，因此两个数列各自的 LIS 之和作为答案上界是可以取到的。

问题转化为对于其中某个数列，删除代价最小的数使得 LIS 长度减小。考虑拆点最小割，对于 DP 值为 1 的点，由 $S$ 到它连边，容量无穷；对于 DP 值为 LIS 的点，向 $T$ 连边，容量无穷；对于 $i < j, a_i < a_j$ 且 $f_i + 1 = f_j$，$i$ 向 $j$ 连边，容量无穷。然后求 $S$ 到 $T$ 的最大流即可。

## 8 实时导航

将每个点拆成 5 个点，分别表示还需等待多久才能到达这个点，那么每次可以通过 BFS 求出 $S$ 到每个点的最短路。

注意到 BFS 过程中不需要访问之前走过的点，所以用 bitset 维护未访问过的点，与当前点的出边表求交，然后遍历其中所有的 1 即可。

时间复杂度 $O(\frac{qn^2}{w})$。

## 9 赌博游戏

首先可以发现最优解可以通过增量法构造。也就是说，对于任意一个 $k$ 个元素的最优解 $A$，存在一个 $k+1$ 个元素的最优解 $B$，满足 $B$ 恰好比 $A$ 多了一个元素。所以我们可以从 $k = 0$ 开始，每次贪心选取会让答案最大的元素加入，作为 $k+1$ 的答案。

证明：

首先因为 $a^2 \geq 4b$，所以对于任意的 $k$，$k^2 + ak + b$ 都非负，这说明它乘上 $w$ 不会有负负得正的情况，使得答案不是凸函数。

然后考虑 $A \oplus B$(即集合异或运算)，令 $d_1, d_2, ..., d_n$ 表示 $A$ 和 $B$ 的区别，其中 $d_i = 0$ 表示第 $i$ 个元素同时属于或同时不属于 $A$ 和 $B$，$d_i = 1$ 表示它只属于 $A$，$d_i = 2$ 表示它只属于 $B$。一共有 3 种可能的情况：

- 有且仅有一个 $d_i \in \{1, 2\}$，这说明 $B$ 是由 $A$ 添加一个元素得到的，显然满足之前的结论。
- 存在某些前缀 $k$，满足 $d_1, d_2, ..., d_k$ 中至少有一个非 0，且 1 和 2 的个数相等。如果这时把 $B$ 中所有 2 扔掉，换成 $A$ 中的 1，那么这不会影响 $i > k$ 部分的元素，故答案不变。
- 最后一个非 0 元素是 1，找到最后一个 $d_j = 2$ 以及最小的 $k$，满足 $k > j$ 且 $d_k = 1$。那么交换 $j$ 和 $k$，答案不会改变，否则 $A$ 和 $B$ 不可能是当前的最优解。

根据上述结论，可以从左往右依次考虑每个元素，维护最优的添加顺序。设 $f_k$ 表示选 $k$ 个元素的答案，$g_k = f_k - f_{k-1}$。因为答案是个凸函数，那么 $g$ 应该满足单调不上升。用平衡树维护最优添加顺序，对于每个点维护它的排名以及 $g$。因为从左往右考虑元素，当前考虑的元素无论 $k$ 是多少，都应该位于子序列的末端，故它的 $g$ 即为 $w(k^2 + ak + b)$，将其插入平衡树中合适的位置后，修改后缀的排名以及 $g$ 即可，通过打标记即可实现。

时间复杂度 $O(n \log n)$。