奇怪的东西

猫

首先来讲讲容斥原理

- 许多人都听说过容斥原理
- 也都记得那个公式
- 然而如果只记下那个公式是不够的
- 我们来想想怎么证明

容斥原理的证明

- 比如我们有n个物品，我们要求出它们<u>**都用上**</u>的方案数，但这个东西不好求
- 接下来我们发现，对于任意一个物品的集合，求出<u>**只用它们（不一定每个都用上）**</u>的方案数是容易的
- 那我们可以用容斥原理了
- 比如$n=3$的情况，共有 $111, 110, 101, 011, 100, 010, 001, 000$ 这8个子集
- 设只用集合s中物品并且都用上的方案数是$f(s)$，不一定都用上的方案数是$g(s)$
- 求一组每个$g(s)$的系数a_s，使最终式子中只有$f(111)$前的系数为1，其他都是0

容斥原理的证明

- $g(s)=\sum_{t\subseteq s} f(t)$，对于集合$t$，设$k=n-|t|$，则包含它为子集的集合$s$有$2^k$个
- 其中，大小为$n-i$的集合有$\binom{k}{i}$（组合数）个，因为可以任选i个为0，其他为1
- 如果定义$a_s=(-1)^{n-|s|}$，对于一个集合t，它的$f(t)$被计算的系数为
- $\sum_{s\supseteq t}(-1)^{n-|s|}=\sum_{i=0}^{k}\binom{k}{i}(-1)^i=\sum_{i=0}^{k}\binom{k}{i}(-1)^i 1^{(k-i)}=(1-1)^k=0^k$
- 对于任何正的k（即是全集的真子集）这个式子都为0
- 全集只在$g(全)=f(全)$被计算1的系数，因此最后式子的结果是$f(全)$达到要求

容斥原理的扩展

- 除了解决“每个都用上”的问题，还能解决“一个都不能用上”或“至少要有一个用上”的问题
- 容斥系数的符号可以自行脑补，容易思考
- 不一定都要枚举子集，如果相同类型的子集的$a_s g(s)$全部一样，可以直接计算
- 这在之后的一些DP和反演中有特殊用途
- 现在来讲几道课题

容斥原理的裸题
ZJOI2016 小星星

- 给出一棵树$T=(V,E)$和一个边集F
- 求有多少个排列p，满足对于任意的$\{x,y\}\in E$，都满足$\{p_x,p_y\}\in F$
- $|V|\le 17$

容斥原理的裸题
ZJOI2016 小星星

- 想到容斥并不是那么容易
- 我们可以这么想
- 求**<u>不一定是排列</u>**的**<u>映射</u>**q的方案数，使得满足$(q_x, q_y) \in F$
- 并且强制要求q_i的值只能从一个集合s里面取
- 如果能解决这个问题，就能枚举子集容斥出每个点都用上（**<u>就是排列</u>**）的方案数

容斥原理的裸题
ZJOI2016 小星星

- 这个问题可以树形DP
- 记$f(i,j)$表示如果$q_i = j$时i子树内的q取值的方案数
- 枚举儿子的状态$f(s,k)$，如果$\{j,k\} \subseteq F$，就令$f(i,j)$加上$f(s,k)$
- 复杂度$O\left(2^{|V|}|V||F|\right)$

容斥原理的裸题
ZJOI2016 小星星

- 这引申出了一种指数级时间复杂度、多项式空间复杂度的排列计数算法
- 我们将排列理解为映射，然后枚举映射的取值集合进行容斥
- 这样有什么用呢？
- 比如我们想要对拍一些哈密顿路径的问题，然而暴力的算法空间开不下
- 那我们就直接容斥即可
- Orz WJMZBMR

容斥原理的裸题
广义错排问题

- 有n个人，每个人手上拿着一个数a_i，这些a_i不一定互不相同
- 求有多少种排列p满足$\forall i, a_i \neq a_{\{p_i\}}$，对质数取模
- $n \leq 5000$

容斥原理的裸题
广义错排问题

- 另一种方向的容斥
- 先把a离散化，设有m个不同的a_i，得到每个a_i的出现次数$c_1 \dots c_m$
- 忽视复杂度，考虑枚举“一定不合法”的位置的子集s
- 则$g(s) = (-1)^{|s|} \frac{(n-|s|)!}{\prod_{i=1}^{m}\left(c_i - i\text{在}s\text{中的出现次数}\right)!}$，这是个多项式系数
- 我们发现对于相同的$|s|$，其容斥系数和分子相同，而分母的每部分也只和大小有关

容斥原理的裸题
广义错排问题

- 设$f(i)=\sum_{|s|=i}\frac{1}{\prod_{i=1}^{m}(i在s中的出现次数)!}=\sum_{|s|=i}\sum_{i=1}^{m}\frac{1}{(i在s中的出现次数)!}$
- 其实就是个背包
- 复杂度$O(n^2)$
- 这题告诉我们，容斥原理不仅仅可以枚举子集，也可以发现性质来优化

容斥原理的裸题
带障碍的网格图路径计数问题

- 给出$R\times C$网格图上面的n个障碍
- 只能向右或向下走
- 求出从$(1,1)$到(R,C)不经过任何障碍的方案数，对质数取模
- $R,C\leq 10^7, n\leq 5000$

容斥原理的裸题
带障碍的网格图路径计数问题

- 如果没有障碍，从(a, b)走到$(a+c, b+d)$的方案数为$\binom{c+d}{c}$种
- 因为你可以任意安排c个向左在总共$(c+d)$步中的位置
- 考虑容斥，枚举经过的障碍的集合，求出“必须但不一定只经过它们”的方案数
- 我们规定集合s有“顺序”$s_1 s_2 \dots s_{|s|}$使得路径满足如果$\forall i < j, s_i$可往右往下到达s_j
- 如果找不出可行的顺序，说明无法一次经过s中的所有节点，$g(s) = 0$
- 考虑记$f(i)$表示$\sum_{s_{|s|}=i}(-1)^{|s|}\frac{g(s)}{i\text{到}(n,m)\text{的方案数}}$，这里除掉方案数为了转移方便

容斥原理的裸题
带障碍的网格图路径计数问题

- 我们考虑在这个集合的末尾添加一个i可达的节点j
- 由于大小+1，因此所有$(-1)^{|s|}$全部取反，即对$f(j)$的贡献为
- (障碍i到障碍j的方案数)$\times\sum_{s_{|s|}=i}(-1)^{|s+1|}g(s) = -$(障碍$i$到障碍$j$的方案数)$\times g(i)$
- 两个障碍的方案数直接用组合数来算，只需要用减法来转移就行了
- 复杂度$O(\max(R, C) + n^2)$
- 很多其他容斥的问题也可以用减法代替加法来转移

有套路的容斥原理
二项式版（二项式反演）

- 有些容斥原理有一些套路来优化复杂度
- 比如如果$g(s)=g(t)$对于所有$|s|=|t|$都成立（其实它们的容斥系数都一样）
- 那么就可以只计算$f(i)$表示大小为i的集合的方案数
- 然后如果有枚举子集求和，可以替换成枚举子集大小j，求$\sum_{j=0}^{i}\binom{i}{j}f(j)$
- 看几个经典问题

有套路的容斥原理
相邻不同的球染色计数

- 求**<u>全部用上</u>**c种颜色染一排n个球的方案，使得相邻两个球颜色不同
- $n \le 10^9, c \le \min(n, 10^7)$

有套路的容斥原理
相邻不同的球染色计数

- 枚举能用的颜色的集合
- 注意到如果能用 $|s|$ 种颜色，方案数为 $|s|(|s|-1)^{n-1}$ （后面每个都只取决于前一个）
- 能用|s|种颜色的集合有$\binom{c}{|s|}$个，容斥系数为$(-1)^{c-|s|}$
- 直接计算每个|s|的贡献即可，用只做质数的技巧可以优化快速幂的速度
- 时间复杂度$O(c\frac{\log n}{\log c})$
- 这就是听起来高端大气上档次的二项式反演！

有套路的容斥原理
DAG子图计数

- 给定有向**<u>完全图</u>**$G = (V, E)$，求有多少个$F \subseteq G$满足它不存在任何环
- $|V| \leq 5000$

有套路的容斥原理
DAG子图计数

- 首先，如果是一个DAG，那么它们必然有一些没有入度的**源点**
- 去掉这些源点后，剩下的图还是DAG，因此这里变成了子问题
- 设$f(s)$表示只考虑s内的点，s的完全子图的边集的DAG子图的方案数
- 我们枚举源点集合t，把边分成2类，然后从$f(s-t)$转移过来
  - 由s中的任何点指向t的边都不能选
  - 由t指向$s-t$的边可选可不选，这样的边有$|t| \times |s-t|$条
- 然而这样t不一定就是源点的集合，有可能还有其他源点，所以容斥

有套路的容斥原理
DAG子图计数

- 这里的容斥其实是求“至少有一个”，将“一个都没有”的符号取反即可
- 由于是完全图，所有的集合的方案数都只和集合的大小有关
- 枚举转移的源点时也只和源点集合的大小有关，最后乘个组合数即可
- 所以就不用枚举子集啦！
- 复杂度$O(n^2)$，似乎可以FFT优化

各种各样的容斥原理

- 我们还有
  - 莫比乌斯反演等数论中的容斥原理
  - 从一个数组容斥得到另一个数组，知g求f
  - 和因数倍数有关的容斥
  - 集合幂级数
- 这里先不讨论
- 现在我们来看一些奇怪的技巧

环状优化问题的小技巧
环的分段问题（YZOJ 2391）

- 现在先来看一道题
- 给出一个长度为n的环，环上面有数字
- 要求将这个环分成若干段，使得每一段的和都不超过m
- 求最小的分段数
- $n \le 10^7$

环状优化问题的小技巧
环的分段问题（YZOJ 2391）

- 我们考虑链的情况，那就是贪心地分段，每满m就分一段
- 那么在环上，我们随便选一个点破环为链，然后进行上述贪心
- 只要这个点在某方案中是分割点（某一段开头），这样贪心出来的结果就是对的
- AKF原本的题解就是多次随机取中点，贪心求解
- 但这个算法并不靠谱，我们考虑一些比较靠谱的算法

环状优化问题的小技巧
环的分段问题（YZOJ 2391）

- 先随便取一个点破环为链，然后给链上的点按顺序标号
- 我们考虑一种方案中，在链上最后一个的切割点p
- 首先p必然满足$\sum_{i=p}^{n} a_i \le m$，否则就必须在后面再分一段
- 然后我们可以从p一直往前贪心分段，直到分割点q_p满足$\sum_{i=1}^{q_p-1} a_i \le m$
- 那么我们发现，还没划分的$p \to n \to 1 \to q_p - 1$这一段一定是不超过$2m$的
- 即剩下这一段的分段不超过两段（为0、1、2段中的一种），设它们的和为s_p
- 那么这种方案的段数，就是p到q_p分的段数 + $\left\lceil \frac{s_p}{m} \right\rceil$

环状优化问题的小技巧
环的分段问题（YZOJ 2391）

- 我们对数组记录前缀和，将求和优化到$O(1)$
- 那么我们现在唯一要做的就是：给定p，快速求出q_p
- 我们先放宽p的定义，不一定需要满足“后面不足一段”的条件
- 首先，如果$\sum_{i=1}^{p-1} a_i \leq m$，那么$p$就是第一个分界点
- 否则，由于限制都是m，设p贪心分出的第一个分界点是$f(p)$，则从$f(p)$开始的是一个相同的子问题！
- 那么p出发的分段就是$p, f(p), f\big(f(p)\big), f\left(f\big(f(p)\big)\right), \ldots, q_p$

环状优化问题的小技巧
环的分段问题（YZOJ 2391）

- $f(p)$可以单调性扫描得到，我们只关心中间分了几次以及最后一个分界点q_p是啥
- 如果我们把这个$f(p)$看做p在一棵树上的父亲，那么这两个值分别代表**根**和**深度**
- 我们是按照每棵树的拓扑序遍历的，根和深度都容易维护，设深度为d_p
- 那么只需要枚举一个合法的p直接计算答案即可
- 复杂度$\Theta(n)$
- 这个方法的妙处在于将原本需要暴力贪心分段的步骤，通过“建立决策点树”的形式$O(1)$求解出我们想要的信息，从而快速稳定的解决了问题

环状优化问题的小技巧
珠链分割（BZOJ 2043）

- 给出一棵带点权的环套树
- 你可以把它划分成k个连通块
- 然后选择一个连通块，将其中在环上的节点的点权平方
- 求一种划分方案使得权值和最小的连通块的权值和最大
- $k \le n \le 10^5, 1 \le a_i \le 1000$

环状优化问题的小技巧
珠链分割（BZOJ 2043）

- 首先能分$k+1$个就能分k个（合并任意相邻两个），其逆否命题也成立
- 容易想到二分答案，设现在判定的是m
- 二分完之后，在树上的部分可以直接贪心地划分，每满m个就分一块
- 最后到环上，每个节点就还有一个附加的值，表示它们的未被分块的连通块点权和
- 我们考虑环上怎么划分
- 平方没啥利于解题的性质，唯一的性质就是一个数平方完不会比原来小
- 那么我们可以枚举平方的是环上的哪一段

环状优化问题的小技巧
珠链分割（BZOJ 2043）

- 但这样枚举复杂度也不对
- 由于贪心地性质，我们应该枚举**极小**的平方段
- 即如果$[l,r]$平方后合法，则不需要枚举$[s,t]\supset[l,r]$
- 所以我们可以枚举r，用单调指针求出l，这样枚举的段数就是线性了
- 现在的任务就是从$l-1$不断往前贪，一直绕过环贪到$r+1$
- 但这样的贪心由于头一直会变，并不容易维护

环状优化问题的小技巧
珠链分割（BZOJ 2043）

- 我们考虑，对这个问题来说，从头贪心分段、从尾贪心分段都是正确的
- 那么我们从头尾两端一起往中间贪心分段，一定也是对的
- 这样就容易维护了，我们维护每个点往左分段前一段的决策点，和往右的决策点
- 这样就是两棵树，那么我们在枚举区间的时候，就可以通过两棵树$O(1)$计算答案
- 时间复杂度$O(n\log \text{树上点权和} + \text{环上点权平方和})$
- 这题的技巧在于将“一端贪到另一端”改成“两端往中间贪”使得信息容易维护
- 现在我们来讲一些CC题

TARPAIR

- 给出一张无向连通图，求有多少种方案，删除两条边使得图不连通
- $n, m \leq 10^6$

TARPAIR

- 首先，如果有c条割边，那么这部分的方案数贡献为$\binom{c}{2} + c(m-c)$
- 接下来就是删除其它边的情况，我们先DFS一棵生成树
- 如果删除的是两条非树边，那么生成树还在，图不会不连通
- 如果删除的只有一条是（非割边的）树边，那么只有当覆盖它的非树边唯一的时候，才能通过删除这条非树边使得图不连通
- 如果删除的两条都是（非割边的）树边，那么只有它们在树上有祖先关系，并且覆盖它们的非树边集合相同时，才能使得它们中间那条链断开（环上随便删两条边）

TARPAIR

- 覆盖的非树边个数容易用树上前缀和计算
- 覆盖非树边的集合可以用哈希来做，不过可能就需要用到奇怪的数据结构
- 我们可以直接对每一条非树边生成一个随机权值，并定义每一条树边的权值是所有覆盖它的非树边的权值的异或和
- 这样，如果一条边的权值为0，则是割边；如果两条边权值不为0且相等，那么可以同时删除它们使得图不连通
- 复杂度$O(n+m+$哈希表的复杂度)

TWOCOMP

- n个城市构成了树形结构
- 有两家铁路公司，每家有若干个铁路的方案，每个方案有一条铁路路径和一个权值
- 从两家的方案中各选出一些方案，使得不存在一个城市被两个公司的铁路同时经过
- 最大化权值之和
- 原版：$n \le 100000$, 每家公司方案的个数$k \le 700$
- 加强版：$n \le 100000$, 每家公司方案的个数$k \le 20000$

TWOCOMP

- 如果A公司的方案i和B公司的方案j有交，那么这两个方案不能同时选中
- 那就是个“二分图带权最大权独立集”问题，转为最小割做
- 在原版数据中，我们可以暴力建图，通过求LCA来判交，然后跑Dicnic
- 但新的数据中，我们就不能这样了，因为每家公司的方案数太多了，建的图的边数可能是平方级别的，空间都开不下

TWOCOMP

- 我们考虑转化一下思路
- 在图的中间放树上的n个点，然后把每个方案都和路径上的每一个点连正无穷边（要根据源还是汇确定每个边的方向）
- 那么现在，在一个割中，一定是每一个点要么把向S（A公司）的边全部割掉，要么把向T（B公司）的边全部割掉，因此这样建图是符合条件的，而且也不需要两两判交，给我们更多的优化空间
- 一个点向链上所有点连边可以用树链剖分套线段树来优化建图
- 复杂度$O\left(\mathrm{MaxFlow}(k+n, k\log^2 n+n)\right)$，Dinic实测可过

ALICE AND BOB

- 给出一棵有点权的有根树，一次游戏在这棵树的一个子树内进行
- A先选择子树内的一个节点，然后B再选择子树内的另一个节点，游戏的得分是这样个节点的点权的异或和
- A希望这个游戏的得分尽量大，而B希望得分尽量小
- 求在每一棵子树上玩的时候，游戏的得分
- $n \le 100000$, 点权 $\le 2^{20}$

ALICE AND BOB

- 题意其实就是：在每一棵子树内，求一个点，使得其他点与它的异或的最小值最大
- 似乎并不是很容易维护？
- 我们考虑，如果已经有了一个集合的Trie树，如何求出这个集合的游戏值
- 我们记录$f(i)$表示Trie树上的i节点（非叶子）的子树内的答案
- 设i的深度为d_i，那么如果这棵Trie子树内不存在二进制下d_i位为1的答案，那么答案就是$\max\left(f(next(i,0)), next(i,1)\right)$
- 什么时候存在二进制下d_i位为1的答案？

ALICE AND BOB

- 如果i的某一棵子树（0或1）中有超过两个数字，那么如果A选择这个子树内的数，B必然也选择这个子树内的数，导致二进制下d_i位为0
- 所以，只有当i的某一棵子树有且仅有一个数字，才可以保留住二进制d_i位为1
- 它的尾数（d_i位以后的数）就直接在另一棵子树内查询异或最小值即可
- 这样的时间复杂度是$O(数的个数\log^2 W)$的
- 我们想办法把它推广到树上

ALICE AND BOB

- 一种思路是Trie树的启发式合并，但这样下来复杂度变成$O(n\log^2 W\log n)$了
- 我们考虑Trie树可以像权值线段树一样合并，并且合并完子树的信息还可以复用
- 那我们用Trie树合并即可
- 于是这个问题就完美解决啦
- 复杂度$O(n\log^2 W)$

完结撒花！

- 谢谢！