Dynamic Programming

sunset

IIIS, THU

July 22, 2020

All Kill

Statement

一场 ICPC 比赛共有 n 个题, 比赛时长为 t 分钟.
第 i 个题在想出来之后需要花费 a_i 的时间来写.
你会在 $b_i \in \{0, 1, 2, \cdots, t-1\}$ 这个时刻想出第 i 个题的做法.
你写题的策略是选择当前会做的编号最小的题目写. 如果在写一个题目的过程中想出了一个编号更小的题目的做法, 你就会不高兴.
求有多少种序列 b_i 让你 AK 且高兴.
$n \leq 10^5$.

All Kill

新增一分钟放在比赛最后，要求比赛的最后一分钟空机，显然这不影响答案.

一个做题时间的分配方案对应序列上的一个区间覆盖方案. 为了便于计数，将序列头尾接在一起变成一个环.

对于空机时间，每分钟视为一个时间段；对于做题时间，将整体视为一个时间段，则一共有 $n + t_e$ 个时间段，其中 t_e 是空机总时长.

按照顺序放每个题. 对于第一题，可以随便选一个位置放；之后的每个题，可以选择覆盖环上的某一段，或是选择放在已有的段里，例如在做第一题的过程中想出第二题. 算出总方案数后，先破环成链，对应的方案数为空机时间个数，再除以环长去掉循环的方案即可.

Amidakuji

Statement

输入整数 n, 构造 $0 \le k \le \lceil \log n \rceil + 1$ 个排列 p_i, 使得对于任意 $0 \le x, y < n$, 都有 $(q_k \circ q_{k-1} \circ \cdots \circ q_1)(x) = y$, 其中 $q_i = p_i$ 或 $q_i = p_i^{-1}$.
$n \le 1000$.

Amidakuji

如果 n 是奇数, 令 $p_{i,j} = j + 2^i \bmod n$ 即可.

如果 $n \bmod 4 = 0$, 构造 $p_0 = (2, 3, 1, 0, 6, 7, 5, 4, \cdots)$, 再套用奇数的构造即可.

如果 $n \bmod 4 = 2$, 构造 $p_0 = (2, 3, 1, 0, \cdots, n-2, n-1)$, $p_1 = (0, 1, 2, \cdots, n-6, n-5, n-2, n-1, n-3, n-4)$ 即可. 注意 $n = 2$ 时无解.

Arcs on a Circle

Statement

给一个长度为 c 的环和 n 个圆弧, 第 i 个圆弧长度为 a_i. 将圆弧随机放在圆上, 求所有圆弧覆盖了整个圆的概率.

$n \leq 6, c \leq 50.$

Arcs on a Circle

为了方便, 取 a_i 最大的圆弧的放置位置的左端点作为原点.

将圆弧的位置分解为整数部分和小数部分. 小数部分的值并不重要, 只用考虑大小关系即可.

枚举 $(n-1)!$ 种不同的大小关系, 对整数部分进行 DP 计算: 一共有 $(n-1)c$ 个不同的位置, 记 $f(i,j,S)$ 表示覆盖前 i 个位置, 当前圆弧最右端点延伸到 j, 用的圆弧集合为 S 的方案数, 简单转移即可.

Classic Towers

Statement

n 个盘子的汉诺塔, 给定初始状态每根柱子上盘子个数 a, b, c, 以及把所有盘子移到一根柱子上的最小步数 k, 构造一组合法的初始状态.

$n \leq 50, k \leq 2^{50}$.

Classic Towers

将问题倒过来, 求从初始局面移动到当前局面的最小步数.

如果最大的盘子位置正确, 则不需要移动.

否则需要花费 2^k 的代价移动这个盘子, 并把其他盘子移动到第三个柱子上.

记 $f(a, b, c, p)$ 表示每根柱子上已经放了 a, b, c 个盘子, 当前所有的盘子在第 p 根柱子是否可行, 简单转移即可.

Clique

Statement

给一个圆的 n 条圆弧, 选出一个尽量大的子集使得其中的圆弧两两有交.
$n \leq 3000$.

Clique

枚举一条必须选的圆弧，使得不存在其他选了的圆弧被它完全包含.

对于与它两端都有交的圆弧，选入不会影响答案；对于与它不相交的圆弧，显然不能选.

剩下与它左端点相交的圆弧和右端点相交的圆弧. 两边的圆弧可以从两个方向接上.

转化为二维平面上的问题：选出若干黑点和白点使得不存在黑点严格在白点的左下方. 线段树优化 DP 即可.

DFS Count

Statement

给一个图, 求不同的 DFS 序个数.

$n \le 18$.

DFS Count

记 $f(S, i)$ 表示已经经过的点集为 S, 当前在 i 点的方案数.

DFS 的回溯过程转移比较困难, 预处理 $g(S, i)$ 表示已经走过的点集为 S, 从 i 出发能走到的点集以辅助 DP.

Evacuation

Statement

有一个长度为 n 的序列 a 和一个常数 s.
q 次询问, 每次给一个区间 $[l, r]$, 询问 $\max_{l \le x \le r} f(l, r, x)$ 的值.
其中 $f(l, r, x)$ 被定义为, 对于所有满足以下条件的序列 b,

- $\sum_{i=1}^{n} b_i = s$;
- $\forall l \le i \le r, b_i \le a_i$;

$\sum_{i=1}^{n} |i - x| b_i$ 的最小可能值.
$n, q \le 2 \times 10^5, 1 < l \le r < n$.

Evacuation

对于一个询问, 可以分成 x 在左边和 x 在右边两种情况. 考虑 x 在右边的情况, 不难发现 l 是不会产生影响的. 记新的函数为 $g(r,x)$.

可以通过预处理 $O(1)$ 回答 $g(r,x)$. 通过 (打表) 观察可得 $g(r,x)+g(r+1,x+1)\geq g(r,x+1)+g(r+1,x)$.

离线处理所有询问. 对于一组询问 $\max_{L\leq x\leq U} g(r,x)$, 将区间 $[L,U]$ 定位在线段树上, 再对线段树每个节点用决策单调性即可.

时间复杂度 $O((n+q)\log n)$ 或 $O((n+q)\log^2 n)$.

Games on DAG

Statement

给一个 n 个点的 DAG, 求有多少个边的子集使得新图上 $SG(1) \neq SG(2)$.
$n \leq 15$.

Games on DAG

记 $f(S)$ 表示只考虑点集 S 时, $SG(1) = SG(2)$ 的方案数.

枚举 $U = \{v \mid SG(v) = 0\}$, 记 $T = S \setminus U$. 考虑各部分的连边情况:

- U 内部不能连边;
- U 和 T 之间可以随意连边, 且每个 T 中的点至少有一条到 U 的边;
- T 内部的连边方案数为 $f(T)$.

简单 DP 即可.

Giant Penguin

Statement

给一个连通图, 其中每个点都在不超过 k 个简单环里.
支持给点打上标记, 或是查询一个点的最近标记点.
$n, m, q \leq 2 \times 10^5, k \leq 10$.

Giant Penguin

随便取一棵生成树，对其点分治.

去掉重心后会生成若干大小不超过原图一半的子树. 这些子树之间会有连边，但不超过 k 条.

包括重心，一共去掉 $k+1$ 个点后，整个图将变成若干个大小不超过原图一半的连通块，递归即可.

对于所有关键点跑 BFS，预处理它们到每个点的距离，修改和查询将非常容易.

Graph Coloring

Statement

给一个 n 个点的竞赛图, 给边 14 染色, 使得对于任意 $i \to j \to k$, $i \to j$ 和 $j \to k$ 不同色.

$n \leq 3000$.

Graph Coloring

注意到 $\binom{14}{7} > 3000$, 可以给每个点一个独一无二的有 7 个 0 和 7 个 1 的 01 串.

对于每条边, 一定有某一位从 0 变成了 1, 将边染色成这个位对应的颜色即可.

Horrible Cycles

Statement

给一个 $2n$ 个点的二分图, 左边第 i 个点向右边前 a_i 个点连边, 求图中的简单环个数.

$n \leq 5000$.

Horrible Cycles

考虑一个一个删点, 每次要么删去右边的一个孤立点, 要么删去左边连向右边所有点的点.

在删点的过程中 DP, 记录左边还有多少个点, 右边的点如果在简单环中则会串起左边的两个点.

Inversions

Statement

求长度为 n, 逆序对为 k 的排列个数.

$n, k \leq 10^5$.

Inversions

从小到大填数, 所求即为 $\sum x_i = k$, $0 \leq x_i < i$ 的解的个数.

容斥, 枚举一些数不符合条件, 则需要求出选出 i 个不同的数和为 j 的方案数.

维护一个序列, 每次整体加一, 并选择是否在最后添加上一个 1. 这样的操作至多 $O(\sqrt{n})$ 次, DP 即可.

注意减掉出现 $n+1$ 的方案数.

Jealous Split

Statement

给一个长度为 n 的序列, 将其划分为 k 部分, 记 s_i 表示第 i 部分的和, m_i 表示第 i 部分的最大值, 构造这样一个划分使得:

$$\forall 1 \le i < k, |s_i - s_{i+1}| \le \max(m_i, m_{i+1}).$$

$n \le 10^5$.

Jealous Split

考虑任意一种不合法的划分，移动划分点使得原来不合法的相邻两部分变得合法。

不难发现，调整之后每段和的平方的和变小了。

因此, 我们只需要找到一种最小化每段和的平方的和的方案即可. 答案是关于 k 的凸函数, 二分斜率之后斜率优化即可.

LIS vs. LDS

Statement

给一个排列, 求一组不相交的 LIS 和 LDS.

$n \leq 5 \times 10^5$.

LIS vs. LDS

一个性质是 LIS 和 LDS 的交不超过 1.

记 f_i 表示经过 i 的 LDS 个数, A 为总的 LDS 个数.

设一个不合法的 LIS 是 $i_1, i_2, \cdots, i_k$, 则 $f_{i_1} + f_{i_2} + \cdots + f_{i_k} = A$.

在求 LIS 的同时维护 f 之和, 只保留两个不同的即可: 一个不合法必然另一个合法. 在模大质数意义下考虑以简化计算.

Match

Statement

给一个小写字母构成的字符串 S, 求一个字典序最小的括号序列 T, 使得对于 T 中任意一对匹配的括号 (i, j), 都有 $S_i = S_j$.
$|S| \leq 10^6$.

Match

维护一个栈, 如果当前栈顶的两个元素相等就弹栈并匹配, 最后如果栈不为空则无解.

结论: 如果 $[l, r]$ 有解且 $[l, y]$ 有解, 则 $[y+1, r]$ 必然有解.

设在最优解中 $(1, p)$ 匹配, 猜测 p 是最大的满足 $S_1 = S_p$ 且 $[2, p-1]$ 有解的数.

证明: 如果有 $p_1 < p_2$ 满足条件, 则 $[p_1, p_2 - 1]$ 一定有解. 如果 1 和 p_1 匹配, 将 p_1 改为左括号可以得到一组更优的解.

另一方面, p 也是最大的满足 $S_1 = S_p$ 且 $[p+1, |S|]$ 有解的数.

记 $f(i, c)$ 表示最大的 j 满足 $[j+1, i]$ 有解且 $S_j = c$, 有转移 $f(i, c) = f(f(i-1, S_i) - 1, c)$ 和 $f(i, S_i) = i$. 化为子问题递归即可.

Policeman and a Tree

Statement

n 个点的树, 有一个警察初始在 s 点, 移动速度为 1; m 个小偷在 x_i 点, 移动速度为无穷大. 如果警察和小偷某个时刻在同一地点, 小偷就会被抓住. 求最坏情况下警察抓住小偷需要的最少时间.

$n, m \leq 50$.

Policeman and a Tree

记 $f(e, a, b)$ 表示警察最后一次走的是 e 这条边, 子树内有 a 个小偷, 子树外有 b 个小偷.

决定小偷在子树内的分配策略时, 做背包 DP: $g(i, j)$ 表示前 i 个儿子分配了 j 个小偷, 让警察花费的最少时间最大是多少.

RPS Robots

Statement

有 n 个人，每个人有一个长度为 k 的石头剪刀布字符串 S.
建一个 n 个点的新图，每个点对应一个人，两个人之间有边当且仅当不管字符串怎么循环移位两个人进行 k 次比赛后的胜场数和负场数不变，求新图中团的个数。
$n \leq 10^5, k \leq 18$.

RPS Robots

对于第一个人, 记 $R = 1, S = \omega, P = \omega^2$;

对于第二个人, 记 $R = 1, S = \omega^2, P = \omega$.

两个人之间连边当且仅当 $A(x)$ 和 $B^R(x)$ 的循环卷积处处相等.

用 DFT 计算循环卷积. 注意到 iDFT 的本质也是求点值, iDFT 后各项系数相等意味着 iDFT 前非常数项系数为 0. 我们只需要求 DFT 后每个位置是否为 0 即可.

设 $B(x) = \sum \frac{1}{b_i} x^i$, 注意到

$$B^R(\omega_k^j) = \sum \frac{1}{b_{k-i-1}} \omega_k^{ij} = \overline{\sum b_{k-i-1} \omega_k^{j(k-i)}} = \overline{\omega_k^j \sum b_i \omega_k^{ij}},$$

即求 $B^R(x)$ 的 DFT 时, 可以直接使用第一组系数.

最后做一个简单的状压 DP 即可.

Search Engine

Statement

给一个长度为 n 的字符串 s, 求一串字符串序列 $\varnothing, t_1, t_2, \cdots, t_n$ 使得 t_{i+1} 是由 t_i 往前或后添加一个字符而成的, 记 $f(t_i, s)$ 表示 t_i 在 s 中的出现次数, 求 $\max \sum f(t_i, s)$.
$n \le 2 \times 10^5$.

Search Engine

建出后缀自动机, 往前加字符就是在 parent 树上不动或者走向某个儿子, 往后加字符就是走转移边.

注意到离开一个节点的时候, 在原串的出现次数不会变多, 所以在当前节点待到最后一定是不劣的. 简单 DP 即可.

Statement

求 $0 \le x_i \le b^i - c$, 且 $\sum_{i=1}^{m} x_i < n$ 的解数.
$m \le 50, 2 \le b \le 10^9, -b+2 \le c \le b-1, 1 \le n < b^{m+1}$.

Simple Counting Problem

一种暴力的想法是容斥, 那么答案为

$$\sum_S (-1)^{|S|} \binom{n - \sum_{j\in S}(b^j - c + 1) + m - 1}{m}.$$

令 $f(x) = \binom{x}{k}$, 将它变成 k 次多项式, 那么由斯特林数容斥得

$$f(x) = \frac{\sum_{i=0}^{k} (-1)^{k-i} \left[{k \atop i}\right] x^i}{k!}.$$

c 的限制不太好算, 直接暴力枚举 $|S|$ 就可以去掉 c 的贡献了.

预处理 $f(i,j,k)$ 表示从 $b^1, b^2, \cdots, b^i$ 中选出 j 个数和的 k 次方之和, 用二项式定理转移. 接下来枚举脱离限制的最高位，之后的数可以任意选，用预处理的 f 来计算。

The Karting

Statement

数轴上有 n 个点, 第 i 个点在 x_i. 选出 m 个互不相同的点并按照一定的顺序排列成环, 其基础收益为走完这个环的距离, 除此以外每次转向 (即 $i < j > k$ 或 $i > j < k$) 也会获得 t 的额外收益. 求最大收益.

$n \leq 100$.

The Karting

记 $f(i, j, k)$ 表示前 i 个点选了 j 个, 构成了 k 条路径的最大收益. 分类讨论一个点的贡献:

- 左进右出或是右进左出, 没有影响.
- 右进右出, 会新增一条路径, 获得收益 $t - 2x_i$.
- 左进左出, 会合并两条路径, 获得收益 $t + 2x_i$.

注意特判最后合上环的转移.

Tree Nesting

Statement

给两棵树 S, T 问 S 有多少个子图与 T 同构。

$|S| \le 1000, |T| \le 12.$

Tree Nesting

枚举 T 的一个根, 在 S 上做树形嵌入.

将树改写成左儿子右兄弟的形式, 记 $f(i,j)$ 表示 S 中 i 的子树和右兄弟表示 T 当前的某个状态的方案数.

转移时枚举对应的 T 中的节点即可. 注意答案还需要除掉 T 自同构的方案数.

Unicyclic Graph Counting

Statement

求 n 个点的基环树个数, 满足第 i 个点的度数为 d_i.

$n \le 10^5$.

Unicyclic Graph Counting

先特判一个大环的情况.

定义一种特殊的 prufer 编码, 只删树不删环, 且 prufer 编码的最后一位必须在环上.

记 $f(i, j, 0/1)$ 表示考虑前 i 个点, 有 j 个在环上, 编码的最后一位是否填过的方案数.

用分治 FFT 优化转移.

Wise Man

Statement

给一个数 x, 一次操作是 $x := (x + f(x)) \bmod m$, 其中 $f(x)$ 为 x 十进制下的最大数位. 求 n 次操作后的结果.

$n, m \le 10^{18}$.

Wise Man

不考虑取模，如果能快速实现 k 次操作后的结果，那问题就好做了：二分出第一次超过 m 的位置，然后暴力取模，直到走完或者成环为止.

对于每个 (i, x, y)，预处理最后 i 位为 $\overline{00\cdots0x}$，前面的位最大值为 y 时，加上 10^i 左右之后走到的位置和走的步数.

对于询问，从低位到高位一步步将最后的位变成 $\overline{00\cdots0x}$ 状，再从高位到低位逐位确定答案即可.

Xor Rank

Statement

由 n 个线性无关向量在 $\mathbb{F}_2$ 上张成的线性空间中, 已知 m 条信息: 线性空间中第 r_i 大的数是 v_i. 求符合要求的线性空间个数.
$n \le 30, m \le 50, v_i \le 10^9$.

Xor Rank

由线性基的性质, 可以先将 r_i 消成主对角线和自由元, 则 r_i 对应的最高位一定是 v_i 的最高位.

记 $f(i,j)$ 表示从小到大考虑前 i 位, 已经填了 j 个基的方案数. 如果这一位不选, 则转移到 $f(i+1,j)$.

如果这一位要选, 先判断是否合法 (是否某个 r 第 j 位为 1 但对应的 v 第 j 位为 0 或相反), 再计算转移系数.

如果这一位是某个限制的最高位, 则必须按照限制放; 否则, 有 2^{i-j} 种方案.

注意特判无解, 无限组解等边界情况.

Yet Another Minimization Problem

Statement

将一个长度为 n 的序列分为 k 段, 一段 $[l, r]$ 的代价是 $l \leq i < j \leq r, a_i = a_j$ 的 (i, j) 对数. 最小化分割代价.
$n \leq 10^5, k \leq 20.$

Yet Another Minimization Problem

记 $f(i,j)$ 表示前 i 个数分为 j 段的最小代价, 不难发现决策单调.

对于区间贡献的计算, 可以使用类似于莫队的方法. 因为决策单调性的存在, 实际上相当于跑了一个双指针.

Contest Strategy

Statement

一场 ICPC 比赛有 n 个题, 第 i 个题要花 a_i 的时间.
你的策略是先随机开 k 个题, 然后写最快的那个题, 写完后如果还有题没开就随机开一道.
求期望罚时.
$n \leq 10^6$.

Contest Strategy

从大到小排序, f_i 表示前 i 个的答案.

对于前 k 个, 他们的系数是固定的, 所以可以直接算贡献.

考虑当前加入第 i 个数的贡献 ($i > k$), 分为两部分: 它自己的贡献和它带来的其他数的系数增加.

先算它自己的贡献, 如果它在前 k 个, 它的系数就是 i, 否则它的系数是一个等差数列, 可以直接算.

再考虑它对别的数的系数的影响, 首先认为别的数没有影响, 然后把少算的系数加上去, 即这个系数原来是 c, 变成了 $c+1$, 我们先算这个 c 再算 1.

Contest Strategy

首先前 $k-1$ 大一定是最后做的, 之后这部分不予考虑.

对于一种情况, 假设这个数的系数是 c, 那么会有 $c-(k-1)$ 个插入 i 的位置使得它的系数增加.

考虑 $f_{i-1}=\sum_{\text{all permutation}}\sum x\times c$, 即所有情况下 x 乘上系数, 那么这里的增量是:

$$\sum_{\text{all permutation}}\sum x\times(c-(k-1))=f_{i-1}-(i-1)!\times\sum x\times(k-1)$$

用前缀和优化一下就可以做到线性.