dp 专题

A_zjzj

Quzhou No.2 High School Zhejiang

April 2025

转移性质

- 对于最值问题的 dp 而言，存在一些无用的转移，或存在一种快速有效的方法找到有用的转移；
- 当我们构建完 dp 后，解决的无非就是快速转移问题；
- 大致方法就是：剔除无用转移、根据转移性质优化枚举量、数据结构优化转移（之后章节会专门讲解）。

CF1476F Lanterns

题目描述

有 n 个灯笼排成一排，第 i 个灯笼具有 p_i 的亮度。每个灯笼要么朝向左，照亮左边编号为 $[i-p_i, i-1]$ 的灯笼，要么朝向右，照亮右边编号为 $[i+1, i+p_i]$ 的灯笼。

找到一种方案，为所有的灯笼确定朝向，使得每一个灯笼被至少一个其他灯笼照亮，或判断无解。

$2 \le n \le 3 \times 10^5, p_i \in [0, n]$。

设计 dp 并尝试转移

- 设 f_i 表示用前 i 个灯笼，最多能够覆盖 $[1, f_i]$ 范围内的灯笼。
- 可以明显发现的转移：
  - i 号位置的灯笼向左：

$$f_i \leftarrow i - 1(\exists 1 \leq j < i, f_j + 1 \geq i - p_i)$$

  - i 号位置的灯笼向右：

$$f_i \leftarrow i + p_i(\exists 1 \leq j < i, f_j \geq i)$$

进一步分析转移

- 但是，这样的转移并不够，还有这种转移：
  - $1 \le j < k < i, f_k + 1 \ge i - p_i$;
  - $f_i \leftarrow j + p_j$;
- 发现 j 的限制和 i 无关，所以在 k 处就能找到一个最优的 j，并在 i 处查询一个最优的 k 即可完成转移。
- 时间复杂度：$O(n \log n)$。

来源不明的一道题

题目描述

给定 n 和 $a_{2\sim n}, b_{2\sim n}$，表示 i 可以一步走到编号在 $[a_i, i-1]$ 中的点，编号在 $[b_i, i-1]$ 中的点可以一步走到 i，求出两两点对间的最短路。

$1 \le n \le 6\times 10^3$，$1 \le a_i, b_i < i$。

思路

- 容易发现，如果 i 先向左走一步到 j，紧接着向右走一步到 k，这是一定不优的，证明如下：
  - 若 $k = i$，显然浪费了两步；
  - 若 $k > i$，则 $b_k \le j < i$，所以 i 能够直接走到 k；
  - 若 $j < k < i$，则 $a_i \le j < k$，所以 i 也能够直接走到 k；
- 如此，最优方案一定是先向右走若干步，再向左走若干步。

转移

- 枚举起点 s，再分两步进行转移。
- 设 f_i 表示 s 到 i 的最短路。
- 向右的转移：$f_i \leftarrow f_j + 1(a_i \le j < i)$。
- 向左的转移：$f_i \leftarrow f_j + 1(b_j \le i < j)$。
- 显然可以用线段树做到 $O(n^2 \log n)$。

寻找性质优化

- 以向右的转移为例，设 $g_x = \max\limits_{j<i \wedge f_j = x} \{j\}$。
- 那么，只需从 $x = f_{i-1} + 1$ 开始，每次将 x 减小 1，直到 $g_{x-1} < b_i$ 停下。
- 为什么正确？只需保证 g 的单调性，而 $g_0, g_1, \cdots, g_{f_{i-1}}$ 的确是单调的，更大的就不一定单调了。

寻找性质优化

- 向左转移也是一样的，重新设 $g_x = \min_{i<j \wedge f_j = x} \{a_j\}$。
- x 也从 $\min\{f_i, f_{i-1}+1\}$ 开始，每次减 1，直到 $g_{x-1} > i$ 停下。
- 单调性同样只在 $g_0, g_1, \cdots, g_{f_{i+1}}$ 满足。

特殊的一类 dp

问题概述

- 这类问题大概长这样：
- 求一个排列 $p_{1\sim n}$，最小（大）化如下值：

$$\sum_{i=1}^{n-1} f(p_i, p_{i+1})$$

- 其中 $f(i,j)$ 如下：

$$f(i,j) = \begin{cases} a(i)+b(j) & i<j \\ c(i)+d(j) & i>j \end{cases}$$

解法

- 考虑按照 p_i 的大小，从小到大插入序列的过程，维护若干连续段。
- 每次插入，大致有一下几种情况：
  - 将两个连续段合并成一个；
  - 插入一个连续段的左边/右边；
- 此时，可以轻松计算出插入元素产生的贡献。另外，需要维护连续段个数，确保最终连续段都合并为一个。
- 可能需要判断插入时是否插在全局的开头或末尾。

CF704B Ant Man

题目描述

- 有 n 个元素，第 i 个元素有五个参数 $(x_i, a_i, b_i, c_i, d_i)$;
- 你需要求出一个 $1 \sim n$ 的排列 p，满足 $p_1 = s, p_n = e$，同时最小化这个排列的权值;
- 一个排列的权值为 $\sum_{i=1}^{n-1} f(p_i, p_{i+1})$，其中 $f(i, j)$ 的值有两种情况:
  - 若 $i > j$，则 $f(i, j) = x_i - x_j + c_i + b_j$;
  - 若 $i < j$，则 $f(i, j) = x_j - x_i + d_i + a_j$;
- $2 \leq n \leq 5 \times 10^3$，$s \neq e$，$1 \leq x_1 < x_2 < \cdots < x_n \leq 10^9$，$1 \leq a_i, b_i, c_i, d_i \leq 10^9$。

Solution

- 这类问题的入门：
  - $a(i) = d_i - x_i$;
  - $b(i) = x_i + a_i$;
  - $c(i) = x_i + c_i$;
  - $d(i) = b_i - x_i$;
- 直接套用这种方法解决即可。

[JOI Open 2016] 摩天大楼

题目描述

将互不相同的 N 个整数 $A_1, A_2, \cdots, A_N$ 按照一定顺序排列。

假设排列为 $f_1, f_2, \cdots, f_N$，要求:
$|f_1 - f_2| + |f_2 - f_3| + \cdots + |f_{N-1} - f_N| \leq L$。

求满足题意的排列的方案数对 $10^9 + 7$ 取模后的结果。

$1 \leq N \leq 100$，$1 \leq L \leq 1000$，$1 \leq A_i \leq 1000$。

Solution

- 此类型 dp 的一个变种，并无多少差别，需要注意权值的计算不能再使用减法了。

dp 构建

- 对于一些具有明显 dp 倾向的题目，dp 构建往往是不难的；
- 但是对于一些题目，dp 的构建往往是问题的关键；
- 另外，dp 构建的顺序同样至关重要，这在一些题目中的表现尤为突出。

The 2023 ICPC Asia East Continent Final Contest A. DFS Order 4

题目描述

对于所有点数为 n，每个点父亲编号小于自身编号的有根树，找到其唯一的最小字典序的 DFS 序，求出该 DFS 序有多少种可能，对 P 取模。

$1 \le n \le 800$，$10^8 \le P \le 1.01 \times 10^9$，$P$ 为质数。

思路

- 考虑给定一个 DFS 序，找到一棵与之对应的树。
- 我们容易得到如下贪心过程：考虑依次遍历 DFS 序中的每个点，维护当前点到根的路径（类似构建虚树的过程）；在加入一个点 u 时：
  - 若栈顶 v 小于 u，则直接连边 $fa_u = v$；
  - 否则需要弹出所有编号大于 u 的点，因为这些点不可能成为 u 的祖先；另外，还需要弹出当前的栈顶，因为该 DFS 的字典序最小，因此儿子是按照编号依次遍历的，所以需要有一个编号小于 u 的点作为 u 的兄弟。最后再连边 $fa_u =$ 栈顶。
- 容易证明，一个 DFS 满足条件当且仅当可以顺利执行上述过程。

转化判定方式

- 于是，我们考虑对于上述贪心方法生成的树 dp。
- 即有如下两点限制：
  - 每个点的父亲编号小于自身编号；
  - 若 u 存在两个儿子 v_1, v_2，则 v_1 儿子中的编号最大值大于 v_2。

简化问题

- 考虑将编号的大小关系建成一张图，容易发现这个 DAG 并不是很好 dp。
- 所以，考虑对于第二条限制容斥，不考虑第二条限制的贡献为 $+1$，考虑第二条限制的反面的贡献为 -1。
- 此时，重新将编号的大小关系建图，并去除无用边，发现此时 DAG 成为了一棵树。
- 而对于树的拓扑序为：$n!\prod\frac{1}{siz_i}$。

构建 dp

- 这部分比较抽象，需要大量画图解决，此处略去。
- dp 方程为：

$$f_{i,j} = \frac{1}{i}\Bigg(f_{i-1,j-1} - f_{i-1,j+1} + \sum_{k=1}^{i-1} f_{k,1} f_{i-1-k,j-1} + \sum_{k=1}^{i-2} f_{k,1} f_{i-1-k,j}\Bigg)$$

- 边界条件：$f_{0,0} = 1$。答案即为 $(n-1)!f_{n-1,1}$，注意特判 $n = 1$ 的情况。时间复杂度：$O(n^3)$，空间复杂度：$O(n^2)$。

3rd Ucup Stage 7 C. Price Combo

题目描述

有 n 个物品，第 i 个物品在商店 A,B 的价格分别为 a_i, b_i。当你在同一个商店中购买两个物品时，只需支付较贵的一个，求出 n 个物品各买一份的最小金额。

$1 \le n \le 2 \times 10^5$。

构建 dp

- 这个题看上去一点没有 dp 的样子，大致有几种方向：网络流、(反悔) 贪心、dp。
- 但首先都需要找到一些性质。
- 我们先假设 a_i 互不相同、b_i 互不相同。
- 这里的性质很简单：若 $a_i < a_j, b_i > b_j$，那么一定不可能出现 i 在 B 购买，j 在 A 购买的情况。
- 把 (a_i, b_i) 看成平面中的一个点，那么 A,B 会形成一条折线作为分界线，所以我们就可以对折线进行 dp。

构建 dp

- 当折线向右走一步时，将下方的点加入 B；当折线向上走一步时，将左边的点加入 A。
- 至于加入计算的贡献，只需要考虑排名的奇偶性。
- 而若 a_i 或 b_i 存在相同的值时，只需要人为设置一下大小关系即可。
- 此时复杂度为 $O(n^2)$。

优化 dp

- 我们枚举 dp 的其中一维，另一维使用线段树维护。
- 具体地，我们需要维护【【每个高度区间的折线】在当前列走到区间右端点的高度】的最优值，可以用线段树维护当前列右侧每个区间加入 B 的贡献。
- 修改时，只需要修改当前列的点对应的折线。

dp 套 dp

- dp 套 dp 的含义就是：在外层 dp 中记录对应内层 dp 的所有状态；
- 另一种理解方式：对于内层 dp 可以理解为一个自动机，外层 dp 就可以看成在自动机上走 k 步到达 u 的信息；
- 对于内层 dp 来说，有经典的 LIS/LCS、字符串匹配等模型。

LIS/LCS 的处理方法

- 以 LIS 为例，朴素的求解 LIS 的 dp 状态为 $f_{i,j}$ 表示前 i 个数，最后一个值 $\le j$ 的 LIS。
- 若使用在 dp 套 dp 中，我们需要对于一个 i，记录所有 $f_{i,j}$ 的值，考虑如何精简状态。
- 我们发现 $f_{i,j} \le f_{i,j+1}$，以及 $f_{i,j+1} \le f_{i,j}+1$。
- 所以将 $f_{i,j}$ 差分后会变成一个 01 序列，只需记录该值即可。
- 而 LCS 同理，差分后同样是 01 序列，可见此处。

[TJOI2018] 游园会

- 只需要把 LCS 和匹配 `NOI` 子串的状态都记录下来即可，时间复杂度：$O(n2^k)$。

3rd Ucup Stage 7 A. Bus Analysis

题目描述

数轴上给定 n 个点 $t_1, t_2, \cdots, t_n$。你可以花费 2 的代价覆盖 $[x, x+20)$ 内的点，也可以花费 6 的代价覆盖 $[x, x+75)$ 内的点。

现在，这 n 个点有可能会消失，请你对于所有 2^n 种可能，求出覆盖所有点的最小代价。并输出所有情况的最小代价之和，对 10^9+7 取模。

$1 \le n \le 10^3, 1 \le t_i \le 10^9, t_i < t_{i+1}$。

以这道题为例，讲解一下 dp 套 dp 的基本解题步骤。

Step1：解决内层 dp

- 什么是内层 dp，就是给定 $t_{1\sim n}$，求出最小的覆盖代价。
- 首先肯定要将 t 排序，然后设 f_i 表示覆盖前 i 个点的最小代价。
- 然后从前往后转移即可。

Step2：压缩内层 dp

- 若以 t_i 为右端点，长度为 75 最多能覆盖到 t_j，则 dp 需要我们记录 $f_j, \cdots, f_{i-1}$ 才能求出 f_i。
- 能否压缩，似乎很难压缩，这说明还有性质没用。
- 什么性质没用？代价为 $2, 6$ 的性质，显然把代价当成 $1, 3$ 不会有影响，最后乘以二即可。
- 若 $x, y \in [j, i-1]$，则 $f_x \le f_y + 3$ (考虑直接在 y 上多覆盖一个 75)。
- 这说明 $f_j, \cdots, f_{i-1}$ 的有效值只有三个。

Step2：压缩内层 dp

- 所以，我们对于一个 i，只需要记录：
  - w：表示 f_i；
  - x：表示最大的满足 $f_x+1=f_i$ 的 x；
  - y：表示最大的满足 $f_y+2=f_i$ 的 y；
  - z：表示最大的满足 $f_z+3=f_i$ 的 z；
- 发现 $w\le n$，$x,y,z\in[i-75,i]$，状态数是 $O(75^3n)$ 的。

Step3：嵌套外层 dp

- 设 $g_{i,w,x,y,z}$ 表示在 i 个点，内层 dp 的状态为 (w,x,y,z) 的方案数。
- 最终，求一下所有的方案的 w 之和即可。
- 然而，我们实际上不需要记录 w，我们在 $g_{i,x,y,z}$ 处记录所有的方案数，以及所有方案中 w 的和，当 w 加一时，更新一下 w 的和即可。
- 时间复杂度：$O(75^3 n)$，常数小，可以通过。

针对此题的优化

- 传统的 dp 套 dp 都是外层 dp 一步一步走的，我们尝试打破这个规矩。
- 我们发现，浪费的部分正是 z 这一维，这一维的唯一作用只有判断增加一个 75 能否覆盖到 i，我们不如直接快进一下，到最大的 i 使得不增加代价也能够覆盖到。
- 此时 f_i 的值并没有改变，且 y 和 z 的值也不变，所以这时候可以把 z 删了。

针对此题的优化

- 具体地，我们设 $g_{i,x,y} = (\sum 1, \sum w)$ 表示覆盖了前 i 个，t_{i+1} 在不增加权值的情况下覆盖不到了。
- 转移时，我们只需要推进一下 i，转移到最后面的一个 i 就行。
- 时间复杂度降为 $O(75^2 n)$。

关于 dp 套 dp

- 很多题目并不像这道题一样通过推性质就能够解决，而是进行一个搜索，发现内层 dp 的状态数很少，然后就做完了。
- 我认为这种题目非常无聊，纯粹是为了考察 dp 套 dp 而设计，完全不如刚刚的那道题那样可以带来很多感受、启发。

dp 技巧

- dp 技巧的范围比较广泛，此处主要讲解一些小 trick;
- 例如：不算重的技巧、使用自动机思想优化 dp 的思想等。

[AGC013D] Piling Up

题目描述

一开始有 n 个颜色为黑白的球，但不知道黑白色分别有多少，m 次操作，每次先拿出一个球，再放入黑白球各一个，再拿出一个球，最后拿出的球按顺序排列会形成一个颜色序列，求颜色序列有多少种。答案对 10^9+7 取模。

$1 \le n, m \le 3 \times 10^3$。

题型分析

- 像如此这样的题目，你稍微好计算一点的是所有可能（初始局面不同也认为不同)。
- 然而，它却只提取了一部分特征，让你计算该特征的方案数。
- 你就需要设计一个“自动机”，能够判断一个方案是否是满足该特征的方案中需要计算到贡献的唯一的那一个。
- 当然，也有可能是容斥、多项式等做法。

题目分析

- 而对于这题来说，总的方案就是从初始局面开始每一步取出/放入后黑球个数构成的序列。
- 而特征就是该序列的差分（表示每一步变化前后的区别）。
- 于是，我们设计的“自动机”为：该序列必须要存在 0 时才计算到贡献。

题目分析

- 我们逐步减少黑球个数，直到再减少一个该方案就不合法了停止（即保证黑球个数存在一个时刻为 0)。
- 此时特征一样的方案变换之后就成为了同一个方案。
- 所以，在此基础上进行 dp 即可。
- 当然，这题也有容斥做法，此处并不讨论。

总结

- 这类问题可以如此总结：给定函数 $A(x,y)=0/1$，求出有多少 x，$\exists y, A(x,y)=1$。
- 做法大致有三种：转化判定方式、带权计数、对判定过程 dp；
- 该题的做法不仅可以理解为转化判定方式，也可以理解为带权计数；
- 更多的资料参考曹立的 2024 国家集训队论文的 3.4 小节。

[BJOI2017] 同构

题目描述

求一个 n 个点的无向简单的不存在非平凡自同构的图最多有多少条边，如果答案不存在，请输出 -1，否则输出答案对 10^9+7 取模的结果。

一个图 G 有非平凡的自同构定义：存在一个 1 到 n 的置换 p 满足对于所有点 u, v，(u, v) 之间有边当且仅当 (p_u, p_v) 之间有边，并且这个置换非平凡也即 $\exists u$，使得 $p_u \neq u$。

$1 \leq n \leq 10^{100}$。

分析性质

- 方便起见，记 $P(G)=0/1$ 表示 G 是否满足条件，即不存在非平凡自同构。
- 那么对于 G 的补图 G'，满足 $P(G)=P(G')$。
- 所以边数的最大值就等于 $\binom{n}{2}$ 减去边数的最小值。
- 而边数最小时，发现 G 一定是一个森林（$n=6$ 除外）。

分析性质

- 首先，我们发现：当 $n = 2, 3, 4, 5, 6$ 是不存在合法的树，其余情况都存在。
- 那么 G 为什么是森林？简要说明：
  - 由于点数为 n 固定，最小化边数就是最小化边数减点数；
  - 而对于一个连通块来说，边数减点数最小只能是 -1，此时是一棵树；
  - 其余不为树的贡献 ≥ 0，不优；
  - 所以，应该塞入尽可能多的树，而剩余的点数只能合并进一个最大的树中，变成一个更大的树（为了保证合法）。

dp 部分

- 至此，我们只需要解决 f_n 表示 n 个点的不存在非平凡自同构的树有几种了。
- 这时，应该如何考虑该 dp？我们先考虑：给定两棵树，如何判断他们是否同构？
- 我们的做法是以重心为根进行树哈希，这提示我们使用重心去重。

dp 部分

- 用 h_n 表示 n 个点的合法有根树的个数，$g_{n,m}$ 表示用 m 个点，组成若干互不同构的大小 $\le n$ 的有根树的方案数。转移：

$$
\begin{aligned}
h_n =& g_{n-1,n-1} \\
g_{n,m} =& \sum_{i=0}^{\min\{h_n,\lfloor \frac{m}{n} \rfloor\}} \binom{h_n}{i} \times g_{n-1,m-n\times i} \\
f_n =& \begin{cases} g_{\frac{n-1}{2},n-1} & 2 \nmid n \\ g_{\frac{n}{2}-1,n-1} + \binom{g_{\frac{n}{2}-1,\frac{n}{2}-1}}{2} & 2|n \end{cases}
\end{aligned}
$$

- f 的转移，考虑一下树的重心的个数，分类讨论一下。

dp 部分

- 但是 f, g, h 的值可能很大，用高精度解决。
- 至于组合数的问题，暴力计算即可。
- 最后，用 python 或高精度跑一下，发现 $n = 266$ 的时候 $\sum_{i=1}^{n} f_i$ 已经超过了 10^{100}。
- 另外，f 的生成函数在 OEIS 中也有，可以用 g 的生成函数表示，但是普通的 dp 显得更为直接。

CF506E Mr. Kitayuta's Gift

题目描述

IOI2020 国家集训队作业。

给定一个小写字符串 s 和一个正整数 n。

要求在 s 中插入恰好 n 个小写字符使其回文的方案数，两个方案不同当且仅当它们得到的串不同，与插入顺序和位置无关。

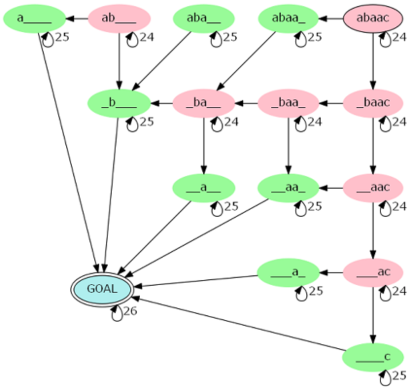
$1 \leq |s| \leq 200$，$1 \leq n \leq 10^9$，答案对 10^4+7 取模。

dp 构建

- 这个题有很明显的 dp 倾向。
- 设 $f_{i,l,r}$ 表示 dp 了最终串的前后 i 个字符，s 从前往后匹配到 l，从后往前匹配到 r 的方案数。
- 转移是简单的，可以做到 $O((n+|s|)|s|^2) \sim O((n+|s|)|s|^2|\Sigma|)$。

自动机思想

将 (l, r) 的转移抽象为一张图：

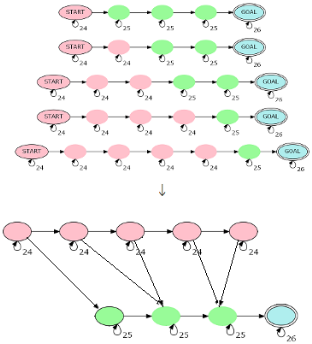


自动机思想

- 容易发现，不同类型的点只有 $24, 25, 26$ 三种，这启发我们对转移的图进行压缩。
- 同时，我们发现，对于一条路径，我们并不在意红点和绿点的出现顺序，调换一下顺序是不影响方案数的。
- 所以，本质不同的状态仅有：(x, y) 表示红点和绿点经过的数量。

自动机思想

于是，我们压缩成这样一张图：



自动机思想

- 这样，状态数就只有 $O(|s|)$ 了，可以使用矩阵快速幂加速。
- 这种使用自动机的思想优化 dp 的方式值得关注。

数据结构优化

- 前面提到的 3rd Ucup Stage 7 C. Price Combo 已经属于数据结构优化，只不过接下来的问题会更倾向于数据结构。
- 另外，[NOI2024] 登山 也是很好的一道数据结构优化 dp 的题，但篇幅有限，不做讲解。

人造情感（emotion）

题目描述

给你一颗 n 个节点的树，以及 m 条路径 (u, v, w)。一个路径集合 S 的重量 $W(S)$ 记为：找出 S 的一个子集满足任何两条路径没有公共点，所有满足条件的子集的路径权值之和的最大值就是 $W(S)$。

记 $f(u, v) = w$ 为最小的非负整数 w，使得对于给定的 m 条边组成的路径集合 U，$W(U \cup \{(u, v, w + 1)\}) > W(U)$ 。计算：

$$\sum_{u=1}^{n} \sum_{v=1}^{n} f(u, v) \mod 998244353$$

$1 \leq n \leq 3 \times 10^5$，$0 \leq m \leq 3 \times 10^5$，$1 \leq w \leq 10^9$。

分析

- $f(u,v)$ 等于 $W(U) - W(U$ 去掉所有和路径 (u,v) 有交的路径 $)$。
- 前者只需要计算一次，而后者需要在原树上删去路径 (u,v) 后计算剩余所有子树的答案，这需要我们计算 u 子树中的答案 f_u 和 u 子树外的答案 g_u。

求解 f_u

- f_u 仅计算路径完全在子树 u 中的答案。
- 那么分类讨论一下 u 是否被一条路径 k 覆盖：
  - 若没有，则 $f_u \leftarrow \sum_{v\in son(u)} f_v$；
  - 若存在 k，则 $\mathrm{LCA}(u_k, v_k) = u$，那么 $f_u \leftarrow w_k + \mathrm{cost}(u_k, v_k)$。
- $\mathrm{cost}(x, y)$ 的计算：
  - 设 $t_u = \sum_{v\in son(fa_u), v\neq u} f_v$；
  - 则 $\mathrm{cost}(x, y) =$ ~~（这个式子太难写了）~~；

求解 g_u

- 和常规的换根 dp 思路一致，自顶向下地推。
- 在 u 时，同样需要找到 $\mathrm{LCA}(u_k, v_k) = u$ 的路径 k。
- 此时的贡献应为：~~(这个式子也太难写了)~~。

计算答案

最后只需要计算一下每个 f_u, g_u 对答案的贡献。

[PA2014] Druzyny

题目描述

将 n 个元素划分为若干非空区间，第 i 个元素有限制：它所在的区间长度不大于 d_i，不小于 c_i，求划分区间个数最大值、以及取到最大值的方案数。

$1 \le n \le 10^6, 1 \le c_i \le d_i \le n$。

dp 构建和性质分析

- 显然有一个线性 dp：f_i 表示前 i 个元素划分的区间个数最大值和方案数，转移如下：

$$f_j \leftarrow f_{i-1} + 1(\max_{x=i}^{j}\{c_x\} \le j - i + 1 \le \min_{x=i}^{j}\{d_x\})$$

- 考虑 j 固定时，$j - i + 1 \le \min_{x=i}^{j}\{d_x\}$ 是单调的，所以可以预处理 p_j 表示满足该式子的最小的 i。
- 同时，我们们可以发现 p_i 是单调不降的。

转移性质

- 转移改写为：

$$f_j \leftarrow f_{i-1} + 1(\max_{x=i}^{j}\{c_x\} \le j - i + 1 \land p_i \le j < i)$$

- 我们需要处理一下 $\max\limits_{x=i}^{j}\{c_x\}$，考虑使用笛卡尔树处理这东西。
- 那么就需要在笛卡尔树上做分治，若在节点 u 上，左右端点分别为 L_u, R_u。

转移性质

- 转移 $f_j \leftarrow f_{i-1}+1$ 的限制转化为:

$$\max\{L_u, p_j\} \le i \le \min\{u, j-c_u+1\}$$

- 接下来我们就开始分类讨论，拆掉 $\max, \min$。

情况一

$$L_u > p_j \wedge u \geq j - c_u + 1 \Longrightarrow j \leq \min\{k-1, u+c_u-1\}$$

- 其中 k 表示最小的满足 $j \in [u, R_u] \wedge p_j \geq L_u$ 的 j，而对于 i 的限制如下：

$$L_u \leq i \leq j - c_u + 1$$

- 发现 j 向右移动的过程，i 的范围也增大 1，所以用线段树查询出最小的 j 对应的 i 的贡献，然后逐步转移。时间复杂度为 $O(\min\{u-L_u, R_u-u\})$，总复杂度为 $O(n\log n)$。

情况二

$$L_u > p_j \wedge u < j - c_u + 1 \Longrightarrow u + c_u - 1 < j < k$$

- 对于 i 的限制为：$L_u \le i \le u$。
- 直接用线段树区间查询贡献并区间修改到对应位置即可。

情况三

$$L_u \le p_j \Longrightarrow k \le j \le R_u$$

- 对于 i 的限制为 $p_j \le i \le \min\{u, j - c_u + 1\}$，这里有一个隐含条件：$p_j \le u$。
- 我们发现，对于一个 j，只可能存在一个 u 满足 $L_u \le p_j \le u \le j \le R_u$，故可以从 k 开始枚举 j 依次转移即可。

总结

- 至此，这道题就被我们在 $O(n \log n)$ 的时间复杂度内解决了。
- 本题的关键在于使用笛卡尔树 $+$ 全局线段树维护的方式，并借助分类讨论的方式，简化转移的范围。

细节讨论

- 这一类非常考验选手分类讨论能力的 dp 题可能会引起不适;
- 但是如果你从做题开始就进行缜密的分类讨论和逻辑推理，就能够很好地锻炼思维的完整性;
- 这种能力可以有效避免在考场上花费大量时间在假算法上;
- ~~虽然我并不具备这样的能力。~~

2024 “钉耙编程”中国大学生算法设计超级联赛（3）1005. 数论

题目描述

给定长为 n 的正整数序列 $\{a_i\}_{i=1}^n$。

定义不交区间集为若干不交的区间 $[l_1, r_1], [l_2, r_2], \cdots, [l_k, r_k]$ 的集合，其中所有元素 $[l_i, r_i]$ 满足 $1 \le l_i \le r_i \le n$。

称一个不交区间集为好的，当且仅当:

$$\gcd_{i=l_1}^{r_1}\{a_i\} = \gcd_{i=l_2}^{r_2}\{a_i\} = \cdots = \gcd_{i=l_k}^{r_k}\{a_i\}$$

对于每个 $x = 1, 2, \cdots, n$，求出有多少个好的不交区间集，存在 $[l_i, r_i]$ 包含 x，对 998244353 取模。

$1 \le n \le 10^5$，$1 \le a_i \le 10^9$。

思路

- ~~这道题重点并不在 dp 上。~~
- 首先，先使用经典结论，在固定左端点 l ，右端点向右的过程中，$\gcd_{i=l}^{r}\{a_i\}$ 只会改变 $O(\log V)$ 次。
- 则外层枚举所有区间 gcd 的值 v，找到所有三元组 (l, r_1, r_2) 表示所有 $[l, r_1], [l, r_1+1], \cdots, [l, r_2-1]$ 的 gcd 都为 v。
- 接下来肯定要先将所有 l, r_1, r_2 离散化。
- 方便起见，接下来的 l, r_1, r_2 都为离散化之后的值，离散化后为 $i = 0, 1, \cdots, k-1$ 的原值为 w_i，不妨令 $w_k = n + 1$。

做法

- 考虑对答案贡献的三元组 (l, r_1, r_2)，容易发现，对于相同的 $i \in [r_1, r_2)$，所有 $r \in [w_i, w_{i+1})$，区间 $[w_l, r]$ 选入好的不交区间集的方案数相同。
- 故设 f_i 为区间右端点小于 w_i 的 $\gcd = v$ 的不交区间集的方案数；类似地，设 g_i 为区间左端点不小于 w_i 的 $\gcd = v$ 的不交区间集的方案数。
- 则对于所有 $j \in [w_l, r]$，$ans_j \leftarrow f_l \times g_{i+1}$。

计算答案

- 于是，考虑计算对于答案数组 ans 的差分 ans'，而其中一部分贡献，对于 $[w_i, w_{i+1})$ 是相同的，所以考虑计算出 b_i 表示对于 $ans'_{w_i+1 \sim w_{i+1}}$ 的贡献。
- 则对于所有 $ans'_{w_l} \leftarrow f_l \times g_{i+1}, b'_{i+1} \leftarrow -f_l \times g_{i+1}$。
- 所以，对于三元组 (l, r_1, r_2)，用前缀和、差分求出：

$$ans'_l \leftarrow f_l \times \sum_{i=r_1}^{r_2-1} g_{i+1} \times (w_{i+1} - w_i)$$

$$b_i \leftarrow -f_l \times g_{i+1} \qquad i \in [r_1, r_2)$$

求解 $\{f_i\}_{i=0}^{k}$

- 边界条件：$f_0 = 1$。考虑从小到大枚举 i，求出 f_i。
- 当求出 f_i 时，枚举左端点落在 i 的三元组 (i, r_1, r_2)，考虑其对于 f 的贡献：

$$f_{j+1} \leftarrow (w_{j+1} - w_j) \times f_i \quad j \in [r_1, r_2)$$

- 这里对于 f_{j+1} 的贡献是上一个区间右端点落在 $[w_j, w_{j+1})$ 的情况，对于上一个区间右端点 $< w_j$ 的情况，增加转移：

$$f_i \leftarrow f_{i-1} \quad i \in (0, k]$$

- 只需差分即可轻松解决。

求解 $\{g_i\}_{i=0}^k$

- 边界条件：$g_k = 1$。考虑从大到小枚举 i，求出 g_i。
- 在求 g_i 时，对于左端点 $\geq w_{i+1}$ 的情况，有转移：

$$g_i \leftarrow g_{i+1} \quad i \in [0, k)$$

- 对于左端点为 w_i 的情况，枚举左端点落在 i 的三元组 (i, r_1, r_2)，考虑其对于 g_i 的贡献，使用前缀和即可。

$$g_i \leftarrow \sum_{j=r_1}^{r_2-1} g_{j+1} \times (w_{j+1} - w_j)$$

总结

- 至此，本题已经完全解决。
- 本题的难点在于，离散化后出现的一系列系数，需要考虑清楚。
- 通过合理的设置前后缀和/差分的方式，使得不会访问到数组未定义的位置。

结语

感谢聆听