

DP选讲

dy0607

雅礼中学

September 30, 2018

Preface

dp相关的内容很多，但有些内容比较套路且不太可能在NOIp出现（例如凸优化/四边形不等式），这里不会涉及。

Preface

dp相关的内容很多，但有些内容比较套路且不太可能在NOIp出现（例如凸优化/四边形不等式），这里不会涉及。

由于时间关系，这里选了一些不需要太多前置知识（例如FFT/FWT）的题，难度大概为提高至省选-。

Preface

dp相关的内容很多，但有些内容比较套路且不太可能在NOIp出现（例如凸优化/四边形不等式），这里不会涉及。

由于时间关系，这里选了一些不需要太多前置知识（例如FFT/FWT）的题，难度大概为提高至省选-。

题目难度与顺序无关。



CF 500 Div1 C

一个长度为 n 的序列 A ，我们称一个元素是好的，当且仅当它严格大于相邻的元素。你可以进行若干次操作，每次将一个元素减1。

对于每个 $k \in [1, \lceil \frac{n}{2} \rceil]$ ，求至少要进行多少次操作使得序列中至少有 k 个好的元素。

$$n \leq 5000, A_i \leq 10^5$$



稍作观察发现，如果最终方案中一个位置是好的，那我们一定不会对它做操作；如果不是，它最终的值是 $A_{i-1} - 1, A_i, A_{i+1} - 1$ 中的一个。



稍作观察发现，如果最终方案中一个位置是好的，那我们一定不会对它做操作；如果不是，它最终的值是 $A_{i-1} - 1, A_i, A_{i+1} - 1$ 中的一个。

$dp(i, j, 0)$ 表示前 i 个元素有 j 个是好的，且已经钦定 A_i 是好的，此时对前 i 个元素至少要进行的操作次数。转移到 $i + 1$ 时需要确保操作后 $A_{i+1} < A_i$ 。



稍作观察发现，如果最终方案中一个位置是好的，那我们一定不会对它做操作；如果不是，它最终的值是 $A_{i-1} - 1, A_i, A_{i+1} - 1$ 中的一个。

$dp(i, j, 0)$ 表示前 i 个元素有 j 个是好的，且已经钦定 A_i 是好的，此时对前 i 个元素至少要进行的操作次数。转移到 $i + 1$ 时需要确保操作后 $A_{i+1} < A_i$ 。

类似的， $dp(i, j, 1/2/3)$ 则表示 A_i 不是好的时的三种情况。 $O(n^2)$ 。



CF 429 Div1 C

一个长度为 n 的序列 A ，定义一个1到 n 的排列 p 是合法的，当且仅当 $\forall i \in [1, n-1], A_{p_i} \times A_{p_{i+1}}$ 不是完全平方数。

求有多少合法的排列，对 $10^9 + 7$ 取模。

$$n \leq 300, A_i \leq 10^9$$



对于每个元素去掉它的平方质因子，问题转化为有多少排列 p 满足 $\forall i \in [1, n-1], A_{p_i} \neq A_{p_{i+1}}$ ，即相邻元素不同。



对于每个元素去掉它的平方质因子，问题转化为有多少排列 p 满足 $\forall i \in [1, n-1], A_{p_i} \neq A_{p_{i+1}}$ ，即相邻元素不同。

先统计有多少种不同的元素，以及每种元素的个数。考虑每次将值相同的所有元素加入排列后的序列。



对于每个元素去掉它的平方质因子，问题转化为有多少排列 p 满足 $\forall i \in [1, n-1], A_{p_i} \neq A_{p_{i+1}}$ ，即相邻元素不同。

先统计有多少种不同的元素，以及每种元素的个数。考虑每次将值相同的所有元素加入排列后的序列。

设 $dp(i, j)$ 表示，已经将前 i 种元素加入序列，此时有 j 对相邻位置相同的方案数。转移时枚举将第 $i+1$ 种元素加入后会将多少对原来不合法的相邻位置拆开，以及会新增多少不合法的相邻位置即可。



对于每个元素去掉它的平方质因子，问题转化为有多少排列 p 满足 $\forall i \in [1, n-1], A_{p_i} \neq A_{p_{i+1}}$ ，即相邻元素不同。

先统计有多少种不同的元素，以及每种元素的个数。考虑每次将值相同的所有元素加入排列后的序列。

设 $dp(i, j)$ 表示，已经将前 i 种元素加入序列，此时有 j 对相邻位置相同的方案数。转移时枚举将第 $i+1$ 种元素加入后会将多少对原来不合法的相邻位置拆开，以及会新增多少不合法的相邻位置即可。

总元素个数为 $O(n)$ ，因此复杂度最坏为 $O(n^3)$ 。



NOI2009 诗人小G

有一个长度为 n 的序列 A 和常数 L, P ，你需要将它分成若干段，每一段的代价为 $|(\sum A_i) - L|^P$ ，求最小代价的划分方案。

$$n \leq 10^5, 1 \leq P \leq 10$$

$$dp(j) = \min_{i=0}^{j-1} |sum_j - sum_i - L|^P + dp(i)$$



$$dp(j) = \min_{i=0}^{j-1} |sum_j - sum_i - L|^P + dp(i)$$

作为一名熟练的OI选手，相信大家一眼就能看出这个方程具有决策单调性。

决策单调性是指，对于任意 $u < v < i < j$ ，若在 i 处决策 v 优于决策 u ，则在 j 处必有决策 v 优于决策 u 。



$$dp(j) = \min_{i=0}^{j-1} |sum_j - sum_i - L|^P + dp(i)$$

作为一名熟练的OI选手，相信大家一眼就能看出这个方程具有决策单调性。

决策单调性是指，对于任意 $u < v < i < j$ ，若在 i 处决策 v 优于决策 u ，则在 j 处必有决策 v 优于决策 u 。

用一个栈维护每个决策更新的区间，新加入一个决策时可以二分得到它的区间， $O(n \log n)$ 。



$$dp(j) = \min_{i=0}^{j-1} |sum_j - sum_i - L|^P + dp(i)$$

作为一名熟练的OI选手，相信大家一眼就能看出这个方程具有决策单调性。

决策单调性是指，对于任意 $u < v < i < j$ ，若在 i 处决策 v 优于决策 u ，则在 j 处必有决策 v 优于决策 u 。

用一个栈维护每个决策更新的区间，新加入一个决策时可以二分得到它的区间， $O(n \log n)$ 。

证明？



本题的证明需要讨论绝对值符号，先考虑里面的值为正的情况，其余的情况类似。



本题的证明需要讨论绝对值符号，先考虑里面的值为正的情况，其余的情况类似。

定义 $f_i(x) = dp(i) + (sum_x - sum_i - L)^P$ 。只需证 $g(x) = f_u(x) - f_v(x)$, $u < v < x$ 单增，也就是随着 x 增大决策 u 相较决策 v 越来越不优。



本题的证明需要讨论绝对值符号，先考虑里面的值为正的情况，其余的情况类似。

定义 $f_i(x) = dp(i) + (sum_x - sum_i - L)^P$ 。只需证 $g(x) = f_u(x) - f_v(x)$, $u < v < x$ 单增，也就是随着 x 增大决策 u 相较决策 v 越来越不优。

$$g(x) = (sum_x - sum_u)^P - (sum_x - sum_v)^P + dp(u) - dp(v)$$

$$g'(x) = P(sum_x - sum_u)^{P-1} - P(sum_x - sum_v)^{P-1}$$

$$g'(x) \geq 0$$



CF 474 Div1+Div2 F

一张 n 个点 m 条边的带权有向图，每条边的长度都为1。求一条最长的路径，满足边权严格递增，且路径上边的顺序与输入中这些边的相对顺序相同。

$$n, m \leq 10^5$$



按输入顺序考虑每一条边。设 $dp(i, j)$ 表示路径到了节点 i ，上一条边的权值 j 时的最长长度。



按输入顺序考虑每一条边。设 $dp(i, j)$ 表示路径到了节点 i ，上一条边的权值 j 时的最长长度。

显然有用的状态是 $O(m)$ 的。我们对于每个点维护一个set来存这些状态以及它们的dp值，并保证dp值是随 j 递增的。这样加入一条边 (u, v) 时，只需要在节点 u 的set上二分就可以进行转移了。同时还需要维护 v 的set。

$$O(n + m \log m)$$



CF 505 Div1+Div2 D

有一棵 n 个点的带权二叉树(并不知道树的形态), 给出对这棵二叉树进行中序遍历得到的权值序列, 判断是否存在与之相符的一棵二叉树, 树上每对相邻节点权值的gcd大于1.

$$n \leq 700, w_i \leq 10^9$$



对于二叉树的一棵子树，其中序遍历是一段连续的区间。一个想法是设 $f(l, r, i)$ 表示是否能将区间 $[l, r]$ 建成一棵以 i 为根的合法二叉树，转移枚举两边子树的根，但这样复杂度为 $O(n^5)$ 。



对于二叉树的一棵子树，其中序遍历是一段连续的区间。一个想法是设 $f(l, r, i)$ 表示是否能将区间 $[l, r]$ 建成一棵以 i 为根的合法二叉树，转移枚举两边子树的根，但这样复杂度为 $O(n^5)$ 。

注意到对于区间 $[l, r]$ 构成的二叉树，除非 $l = 1, r = n$ ，否则它一定是 $r + 1$ 的左子树，或者 $l - 1$ 的右子树。因此我们只关心根节点与 $l - 1$ 和 $r + 1$ 的权值的 gcd 是否为 1，而不需要知道根是哪个节点。



对于二叉树的一棵子树，其中序遍历是一段连续的区间。一个想法是设 $f(l, r, i)$ 表示是否能将区间 $[l, r]$ 建成一棵以 i 为根的合法二叉树，转移枚举两边子树的根，但这样复杂度为 $O(n^5)$ 。

注意到对于区间 $[l, r]$ 构成的二叉树，除非 $l = 1, r = n$ ，否则它一定是 $r + 1$ 的左子树，或者 $l - 1$ 的右子树。因此我们只关心根节点与 $l - 1$ 和 $r + 1$ 的权值的gcd是否为1，而不需要知道根是哪个节点。

于是状态数变为 $O(n^2)$ ，转移仍然枚举根即可， $O(n^3)$



CF 507 Div1 D

给出一棵树，对每个 $k \in [1, n]$ ，求在树上最多能找到多少条不相交的包含 k 个节点的简单路径。

$$n \leq 10^5, TL = 7s$$



对于一个单独的 k ，这是一个简单的树形dp：维护一个全局的答案；对每个点记 $f(v)$ 表示在选尽量多条路径的前提下，节点 v 往下的路径最长是多少。



对于一个单独的 k ，这是一个简单的树形dp：维护一个全局的答案；对每个点记 $f(v)$ 表示在选尽量多条路径的前提下，节点 v 往下的路径最长是多少。

对于某个节点 s ，考虑能否用儿子的 f 凑出一条长度至少为 k 的链，如果能则将答案加1，并令 $f(s) = 0$ ；否则 $f(s) = \max f(child) + 1$ 。



对于一个单独的 k ，这是一个简单的树形dp：维护一个全局的答案；对每个点记 $f(v)$ 表示在选尽量多条路径的前提下，节点 v 往下的路径最长是多少。

对于某个节点 s ，考虑能否用儿子的 f 凑出一条长度至少为 k 的链，如果能则将答案加1，并令 $f(s) = 0$ ；否则 $f(s) = \max f(child) + 1$ 。

设 $ans(x)$ 为 $k = x$ 时的答案，则一定有：

$$ans(x) \leq \lfloor \frac{n}{x} \rfloor, ans(x) \leq ans(x-1)$$



对于一个单独的 k ，这是一个简单的树形dp：维护一个全局的答案；对每个点记 $f(v)$ 表示在选尽量多条路径的前提下，节点 v 往下的路径最长是多少。

对于某个节点 s ，考虑能否用儿子的 f 凑出一条长度至少为 k 的链，如果能则将答案加1，并令 $f(s) = 0$ ；否则 $f(s) = \max f(child) + 1$ 。

设 $ans(x)$ 为 $k = x$ 时的答案，则一定有：

$$ans(x) \leq \lfloor \frac{n}{x} \rfloor, ans(x) \leq ans(x-1)$$

因此 $ans(x)$ 只有 $O(\sqrt{n})$ 种取值，且取值相同的 x 是连续的一段。

直接二分即可，复杂度为 $O(n\sqrt{n} \log n)$



HAOI2018 奇怪的背包

有一个奇怪的背包，背包的重量是放入其中的物品总体积对 P 取模后的结果。

现有 n 种体积不同的物品，第 i 种占用体积为 V_i ，每种物品都有无限个。 q 次询问，每次询问给出重量 w_i ，你需要回答有多少种放入物品的方案，能将一个初始为空的背包的重量变为 w_i 。注意，两种方案被认为是不同的，当且仅当放入物品的种类不同。输出答案对 $10^9 + 7$ 取模的结果。

$$n, q \leq 10^6, P \leq 10^9, 0 < w_i < P$$



题目就是求，有多少种选择物品的方式，使得下面的同余方程有解，（假设选择的物品的体积为 a_1, a_2, \dots, a_k ）：

$$\sum_{i=1}^k x_i a_i \equiv w_j \pmod{P}$$



题目就是求，有多少种选择物品的方式，使得下面的同余方程有解，（假设选择的物品的体积为 a_1, a_2, \dots, a_k ）：

$$\sum_{i=1}^k x_i a_i \equiv w_j \pmod{P}$$

即

$$\sum_{i=1}^k x_i a_i + x_0 P = w_j$$



题目就是求，有多少种选择物品的方式，使得下面的同余方程有解，（假设选择的物品的体积为 a_1, a_2, \dots, a_k ）：

$$\sum_{i=1}^k x_i a_i \equiv w_j \pmod{P}$$

即

$$\sum_{i=1}^k x_i a_i + x_0 P = w_j$$

根据不定方程的知识，我们知道这个方程有解当且仅当 $\gcd(a_1, a_2, \dots, a_k, P) \mid w_j$.



于是我们就可以DP了，设 $dp(i, x)$ 表示前 i 种物品，我们选择的物品的体积的 gcd 再和 P 做 gcd 的结果为 x ，选择的方案数是多少。



于是我们就可以DP了，设 $dp(i, x)$ 表示前 i 种物品，我们选择的物品的体积的 gcd 再和 P 做 gcd 的结果为 x ，选择的方案数是多少。

转移只需要考虑是否选这个数，这样做的复杂度是 $O(n \times d(P))$ 的，其中 $d(P)$ 表示 P 的约数个数。由于约数个数大约是 $P^{\frac{1}{3}}$ 的级别（此题中不超过1500）。



于是我们就可以DP了，设 $dp(i, x)$ 表示前 i 种物品，我们选择的物品的体积的 gcd 再和 P 做 gcd 的结果为 x ，选择的方案数是多少。

转移只需要考虑是否选这个数，这样做的复杂度是 $O(n \times d(P))$ 的，其中 $d(P)$ 表示 P 的约数个数。由于约数个数大约是 $P^{\frac{1}{3}}$ 的级别（此题中不超过1500）。

优化也很简单，对于两个物品，如果他们的体积与 P 的 gcd 是相同的，那么它们造成的影响也是相同的，可以放到一起算。相当于物品种类数其实也是 $d(P)$ 的，于是复杂度变为 $O((d^2(P) + n + q) \log P)$ ， \log 是取 gcd 的复杂度。



AIM Tech 5 Div1+Div2 G

交互题。有一个未知数 $x, x \in [1, M]$ 。你的目标是猜出这个数，为此你可以进行至多5次询问。每次可以询问一个长为 k 的序列 t , $1 \leq t_1 < t_2 < \dots < t_k \leq M$ 。若 $x = t_i$ ，则视为猜测成功；否则交互库会告诉你 x 在这 k 个数划分出的 $k+1$ 个区间的哪一个中。

k 由你决定，但需要保证 $1 \leq k \leq 10^4$ 且 $k \leq x$ ，若你的询问出现了 $k > x$ ，则视为猜测失败。

原题中 M 开到了上界($M = 10,004,205,361,450,474$)，且交互库是adaptive的（ x 并非一开始就固定，而是随你的询问而定）。



假设我们现在只知道 $x \in [L, R]$ ，我们就必须保证询问中 $k \leq L$ 。

如果 $L \geq 10^4$ ，每次询问 $k = 10^4$ 一定最优，此时我们可以方便地构造最优的询问。



假设我们现在只知道 $x \in [L, R]$ ，我们就必须保证询问中 $k \leq L$ 。

如果 $L \geq 10^4$ ，每次询问 $k = 10^4$ 一定最优，此时我们可以方便地构造最优的询问。

否则考虑 dp，设 $dp(L, j)$ 表示当前知道 $x \geq L$ ，还有 j 次剩余的询问时，要保证一定猜测成功， R 最大为多少。



假设我们现在只知道 $x \in [L, R]$ ，我们就必须保证询问中 $k \leq L$ 。

如果 $L \geq 10^4$ ，每次询问 $k = 10^4$ 一定最优，此时我们可以方便地构造最优的询问。

否则考虑 dp，设 $dp(L, j)$ 表示当前知道 $x \geq L$ ，还有 j 次剩余的询问时，要保证一定猜测成功， R 最大为多少。

怎么转移呢？显然我们应该让 $t_1 = dp(L, j - 1) + 1$ ，
 $t_i = dp(t_{i-1} + 1, j - 1) + 1$ 。这样我们也构造出了最优的询问。

有用的状态很少，可以快速通过。



Codeplus 2018.3 白金元首与莫斯科

在一个 $n \times m$ 的网格区域中存在一个陆军单位需要补给，每个格子为空地或障碍物中的一种。每一架运输机可以向两个相邻的空地投放物资，每个格子至多只能得到一次投放。

由于天气原因，陆军单位所在的确切位置并不能确定。对于每个空地格子，求当陆军单位在其中（视作障碍物）时，用若干架运输机向其余空地投放物资的不同方案数，对 $10^9 + 7$ 取模。两个投放方案不同，当且仅当存在一个格子在一个方案中被投放而另一方案中未被投放，或存在两个被投放的格子，在一个方案中被同一架运输机投放而在另一方案中非然。

$$n, m \leq 17$$



容易想到状压dp, $f(i, j, S)$ 表示当前考虑到第 i 行第 j 列, 轮廓线上状态为 S 时的方案, S 表示 $(i, 1) \dots (i, j), (i-1, j+1) \dots, (i-1, m)$ 这些位置是否被占用。转移考虑不放/横着放/竖着放。

这样做一次是 $O(nm2^m)$ 的, 而我们总共需要做 $O(nm)$ 次。



容易想到状压dp, $f(i, j, S)$ 表示当前考虑到第 i 行第 j 列, 轮廓线上状态为 S 时的方案, S 表示 $(i, 1) .. (i, j), (i - 1, j + 1) ..., (i - 1, m)$ 这些位置是否被占用。转移考虑不放/横着放/竖着放。

这样做一次是 $O(nm2^m)$ 的, 而我们总共需要做 $O(nm)$ 次。

注意到每次只有一个格子变化, 一种技巧是从前往后做一次dp, 再从后往前做一次dp, 在变化的位置合并两个dp数组。



容易想到状压dp, $f(i, j, S)$ 表示当前考虑到第 i 行第 j 列, 轮廓线上状态为 S 时的方案, S 表示 $(i, 1) \dots (i, j), (i-1, j+1) \dots, (i-1, m)$ 这些位置是否被占用。转移考虑不放/横着放/竖着放。

这样做一次是 $O(nm2^m)$ 的, 而我们总共需要做 $O(nm)$ 次。

注意到每次只有一个格子变化, 一种技巧是从前往后做一次dp, 再从后往前做一次dp, 在变化的位置合并两个dp数组。

但这里直接合并不太方便, 需要对转移做一些修改。由于问题等价于求用 $1 \times 1, 1 \times 2, 2 \times 1$ 的长方形填满整个网格的方案数, 转移时新增一个用 1×1 填的决策, 并保证放满。这样在合并时, 两边状态是一一对应的, 直接枚举一边的状态 S 即可, 总复杂度为 $O(nm2^m)$ 。

Thanks