Solution

ExfJoe

福建省长乐第一中学

March 7, 2017

Outline

采蘑菇

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案
- 链：每一个种类的贡献独立，因此把同一种类一起处理，相邻两个同一种类点中这一段的答案变化相同，可以通过差分再前缀和的方式求解

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案
- 链：每一个种类的贡献独立，因此把同一种类一起处理，相邻两个同一种类点中这一段的答案变化相同，可以通过差分再前缀和的方式求解
- 种类数只有 3：考虑对于每个种类进行点分治，每次统计经过重心的路径对答案的贡献

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案
- 链：每一个种类的贡献独立，因此把同一种类一起处理，相邻两个同一种类点中这一段的答案变化相同，可以通过差分再前缀和的方式求解
- 种类数只有 3：考虑对于每个种类进行点分治，每次统计经过重心的路径对答案的贡献
- 若点到重心有这个种类，则答案为当前总点数减去重心下它所处子树的点数；否则答案为整个子树中满足到重心有这个种类的点的数量减去它所在子树中满足这个条件的点数

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案
- 链：每一个种类的贡献独立，因此把同一种类一起处理，相邻两个同一种类点中这一段的答案变化相同，可以通过差分再前缀和的方式求解
- 种类数只有 3：考虑对于每个种类进行点分治，每次统计经过重心的路径对答案的贡献
- 若点到重心有这个种类，则答案为当前总点数减去重心下它所处子树的点数；否则答案为整个子树中满足到重心有这个种类的点的数量减去它所在子树中满足这个条件的点数
- 满分做法：实际上一次点分时可以同时统计所有种类的贡献

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案
- 链：每一个种类的贡献独立，因此把同一种类一起处理，相邻两个同一种类点中这一段的答案变化相同，可以通过差分再前缀和的方式求解
- 种类数只有 3：考虑对于每个种类进行点分治，每次统计经过重心的路径对答案的贡献
- 若点到重心有这个种类，则答案为当前总点数减去重心下它所处子树的点数；否则答案为整个子树中满足到重心有这个种类的点的数量减去它所在子树中满足这个条件的点数
- 满分做法：实际上一次点分时可以同时统计所有种类的贡献
- 第一种贡献只需要统计种类个数，第二种贡献可以预先 dfs 一遍统计每个种类的贡献，再 dfs 第二遍统计贡献

采蘑菇

- $n \leq 2000$：n 遍 dfs，统计路径中每个种类出现次数计算答案
- 链：每一个种类的贡献独立，因此把同一种类一起处理，相邻两个同一种类点中这一段的答案变化相同，可以通过差分再前缀和的方式求解
- 种类数只有 3：考虑对于每个种类进行点分治，每次统计经过重心的路径对答案的贡献
- 若点到重心有这个种类，则答案为当前总点数减去重心下它所处子树的点数；否则答案为整个子树中满足到重心有这个种类的点的数量减去它所在子树中满足这个条件的点数
- 满分做法：实际上一次点分时可以同时统计所有种类的贡献
- 第一种贡献只需要统计种类个数，第二种贡献可以预先 dfs 一遍统计每个种类的贡献，再 dfs 第二遍统计贡献
- 也可以从链的情况拓展，同一种类将树分为几个连通块，将这些连通块转为 dfs 序上的区间，离线扫一遍即可

Outline

序列游戏

序列游戏

- 难点在于删除一段后两端会合并，需要找到合适的状态表示

序列游戏

- 难点在于删除一段后两端会合并，需要找到合适的状态表示
- 对于区间 $[l, r]$，若你已经确定 r 要与后面的某 k 个元素连在一起进行删除，那么 $[l, r-1]$ 的元素就和后面的那某 k 个元素没有关系了

序列游戏

- 难点在于删除一段后两端会合并，需要找到合适的状态表示
- 对于区间 $[l, r]$，若你已经确定 r 要与后面的某 k 个元素连在一起进行删除，那么 $[l, r-1]$ 的元素就和后面的那某 k 个元素没有关系了
- 基于这个想法来考虑如下 DP：

序列游戏

- 难点在于删除一段后两端会合并，需要找到合适的状态表示
- 对于区间 $[l, r]$，若你已经确定 r 要与后面的某 k 个元素连在一起进行删除，那么 $[l, r-1]$ 的元素就和后面的那某 k 个元素没有关系了
- 基于这个想法来考虑如下 DP：
- $f_{i,j,k}$ 表示对于区间 $[i, j]$，j 与后面的 k 个一定要连成一个连续段进行删除所能得到的最大分数

序列游戏

- 难点在于删除一段后两端会合并，需要找到合适的状态表示
- 对于区间 $[l, r]$，若你已经确定 r 要与后面的某 k 个元素连在一起进行删除，那么 $[l, r-1]$ 的元素就和后面的那某 k 个元素没有关系了
- 基于这个想法来考虑如下 DP：
- $f_{i,j,k}$ 表示对于区间 $[i, j]$，j 与后面的 k 个一定要连成一个连续段进行删除所能得到的最大分数
- $g_{i,j,k}$ 与 f 一样，但 f 表示的状态是可能有峰 (比相邻两个元素都大) 的，而 g 是没有峰的

序列游戏

序列游戏

- 考虑状态转移，若 j 和后面那 k 个直接消掉：

序列游戏

- 考虑状态转移，若 j 和后面那 k 个直接消掉：
-

$$f_{i,j,k} = f_{i,j-1,0} + v_{k+1}$$

序列游戏

- 考虑状态转移，若 j 和后面那 k 个直接消掉：
- $$f_{i,j,k} = f_{i,j-1,0} + v_{k+1}$$
- 若 j 连着 $[i, j-1]$ 中某些数一起消：

序列游戏

- 考虑状态转移，若 j 和后面那 k 个直接消掉：
- $$f_{i,j,k} = f_{i,j-1,0} + v_{k+1}$$
- 若 j 连着 $[i, j-1]$ 中某些数一起消：
- $$f_{i,j,k} = \max_{i \le p < j, a_p = a_j + 1} \{g_{i,p,k+1} + f_{p+1,j-1,0}\}$$

序列游戏

- 考虑状态转移，若 j 和后面那 k 个直接消掉：
- $$f_{i,j,k} = f_{i,j-1,0} + v_{k+1}$$
- 若 j 连着 $[i, j-1]$ 中某些数一起消：
- $$f_{i,j,k} = \max_{i \le p < j, a_p = a_j + 1} \{g_{i,p,k+1} + f_{p+1,j-1,0}\}$$
- $$f_{i,j,k} = \max_{i \le p < j, a_p = a_j - 1} \{f_{i,p,k+1} + f_{p+1,j-1,0}\}$$

序列游戏

- 考虑状态转移，若 j 和后面那 k 个直接消掉：
- $$f_{i,j,k} = f_{i,j-1,0} + v_{k+1}$$
- 若 j 连着 $[i, j-1]$ 中某些数一起消：
- $$f_{i,j,k} = \max_{i \le p < j, a_p = a_j + 1} \{g_{i,p,k+1} + f_{p+1,j-1,0}\}$$
- $$f_{i,j,k} = \max_{i \le p < j, a_p = a_j - 1} \{f_{i,p,k+1} + f_{p+1,j-1,0}\}$$
- g 的转移也是类似

序列游戏

序列游戏

- 原题不一定要全部消完整个序列，而上面的 DP 是全部消完的，所以我们再利用一个子 DP 计算答案即可

序列游戏

- 原题不一定要全部消完整个序列，而上面的 DP 是全部消完的，所以我们再利用一个子 DP 计算答案即可
- 直接 DP 复杂度为 $O(n^4)$

序列游戏

- 原题不一定要全部消完整个序列，而上面的 DP 是全部消完的，所以我们再利用一个子 DP 计算答案即可
- 直接 DP 复杂度为 $O(n^4)$
- 由于相同的数不超过 7 个，所以预处理一个 *last* 数组表示某个位置之前某个数的最近的出现位置

序列游戏

- 原题不一定要全部消完整个序列，而上面的 DP 是全部消完的，所以我们再利用一个子 DP 计算答案即可
- 直接 DP 复杂度为 $O(n^4)$
- 由于相同的数不超过 7 个，所以预处理一个 *last* 数组表示某个位置之前某个数的最近的出现位置
- 时间复杂度 $O(7 \times n^3)$

Outline

石子游戏

石子游戏

- 与叶子层数同奇偶的称为奇数层，其他称为偶数层

石子游戏

- 与叶子层数同奇偶的称为奇数层，其他称为偶数层
- 容易看出它是个阶梯 NIM 游戏，偶数层没有用，用奇数层所有结点石子数的异或和即可判断胜负

石子游戏

- 与叶子层数同奇偶的称为奇数层，其他称为偶数层
- 容易看出它是个阶梯 NIM 游戏，偶数层没有用，用奇数层所有结点石子数的异或和即可判断胜负
- 知道了判断胜负的方式，求解方案也很简单，枚举每个结点，获胜的移动方式是确定的

石子游戏

- 与叶子层数同奇偶的称为奇数层，其他称为偶数层
- 容易看出它是个阶梯 NIM 游戏，偶数层没有用，用奇数层所有结点石子数的异或和即可判断胜负
- 知道了判断胜负的方式，求解方案也很简单，枚举每个结点，获胜的移动方式是确定的
- 若移动奇数层结点，则移动后要求其他奇数层石子异或和 xor 当前枚举结点剩余石子数为 0，即可以直接得出当前需要剩余多少石子，若可以满足则可以计入答案 (非叶结点方案数为 2)

石子游戏

- 与叶子层数同奇偶的称为奇数层，其他称为偶数层
- 容易看出它是个阶梯 NIM 游戏，偶数层没有用，用奇数层所有结点石子数的异或和即可判断胜负
- 知道了判断胜负的方式，求解方案也很简单，枚举每个结点，获胜的移动方式是确定的
- 若移动奇数层结点，则移动后要求其他奇数层石子异或和 xor 当前枚举结点剩余石子数为 0，即可以直接得出当前需要剩余多少石子，若可以满足则可以计入答案 (非叶结点方案数为 2)
- 偶数层结点类似，枚举它移动到哪个奇数层儿子，进而可以知道需要移动多少个石子，若满足可以计入答案