树形 DP 与状压 DP

房天乐

2024 年 7 月 27 日

扫雷（arbiter）

最朴素的想法是，首先枚举子矩阵边长，然后分别枚举两个子矩阵的右下角坐标，最后判断两个子矩阵是否相等。时间复杂度 $O(n^7)$。

注意到上述方法判断两个子矩阵是否相等单次的复杂度是 $O(n^2)$ 的，考虑二维哈希，预处理出每个左上角矩阵的哈希值，判断相等时直接 $O(1)$ 计算即可。总时间复杂度 $O(n^5)$。

扫雷（arbiter）

我们发现枚举右下角下标的 $O(n^4)$ 复杂度难以承受，而对于确定的子矩阵边长，共有 $O(n^2)$ 个子矩阵。考虑将第一个矩阵中子矩阵的哈希值存入一个哈希表中，计算第二个矩阵的子矩阵哈希值并在哈希表中查询，就可以将总时间复杂度降至 $O(n^3)$。

最后可以发现有公共子矩阵的矩阵边长具有单调性，因此可以将枚举子矩阵边长改为二分边长，时间复杂度 $O(n^2 \log n)$。

失忆症（amnesia）

当 $k = 0$ 时即为求最长上升子序列（LIS），使用二分或树状数组，时间复杂度 $O(n \log n)$。

考虑 $k = 1$ 的做法。注意到此时仅会对序列进行一次改变，考虑在二分求 LIS 时维护两个数组 f_0 和 f_1，其中 $f_x(i)$ 表示当前改变过 x 次后长为 i 的 LIS 的最后一个元素的最小值。那么我们不仅要对每个数组进行二分操作，同时也要维护 f_0 的更新对 f_1 的贡献，即 f_1 对更新后的 f_0 求出改变一个元素拼接后的长度取最大值。注意到枚举序列中的一个位置时，f_0 只会更新一个位置，因此 f_1 也只需要对应更新一个位置即可。时间复杂度 $O(n \log n)$。

失忆症（amnesia）

可以发现 $k=1$ 的做法可以扩展到 $k \le 20$。考虑维护 $(k+1)$ 个数组 f_0 至 f_k，每个数组不仅进行二分操作，同时维护来自上一级的更新。需要注意的是若 $i<j$，则 f_i 的更新在 $(j-i)$ 轮后会影响到 f_j，因此枚举序列中的一个位置时 f_x 需要维护来自上一级的 x 个更新，故单次共需维护 $O(k^2)$ 次更新。根据 x 降序进行更新即可满足无后效性。总时间复杂度 $O(kn\log n+k^2n)$。

塞莱斯特山（celeste）

一个显然的性质是对于区间 $[l, r]$，r 点必须设置一个岗哨。那么有一个朴素的做法是，枚举区间 $[l, r]$，从右向左枚举点，若当前点未被监视则设置岗哨，同时计算它左边的点哪些能被它监视。时间复杂度 $O(n^4)$。

考虑固定右端点 r，倒序枚举左端点 l，记录 $[l+1, r]$ 中 r 能监视到的下标最小的点 p，则 $[l+1, p-1]$ 和 $[p+1, r-1]$ 此时是互相不可见的，因此两边互不影响。此时若 r 看不见 l，则 $f(l, r) = \min(f(l, p), f(l, p-1)) + f(p, r)$；若 r 能看见 l，则 $f(l, r) = \min(f(l+1, p), f(l+1, p-1)) + f(p, r)$，并将 p 更新为 l。

r 能跨过 p 监视 l 的充要条件是 $\dfrac{h_r - h_l}{r - l} > \dfrac{h_r - h_p}{r - p}$。总时间复杂度 $O(n^2)$。

培训（popcount）

当无操作 1 时，注意到操作 2 会使值域 V 缩小至 $\log V$，因此一个数进行 $O(\log^* n)$ 次操作就会 ≤ 1。因此可以用线段树维护序列，若一个区间内元素全部满足 ≤ 1 即打上标记，操作 2 时暴力递归，不进入打上标记的子树即可。时间复杂度 $O(q \log n)$。然而有操作 1 时加法操作会打破势能。注意到当进行一次操作 2 后，整个区间的值域变为 $\log V$。仍然使用线段树维护，可以发现最后一次求 popcount 后，无论进行多少次加法操作，都可以表示为 $\text{popcount}(sth) + b$。因此考虑维护标记：

$$f(\text{popcount}(x + a)) + b$$

培训（popcount）

其中 $f(x)$ 为定义域和值域均为 $O(\log V)$ 的映射。这样做的原因是修改序列可以看作若干 $\text{popcount}(x+b)$ 的复合，它的值域为 $O(\log V)$，而从第二次开始定义域也为 $O(\log V)$。因此我们可以直接计算第一次时函数的样子，从第二次开始暴力进行函数复合，最后的加法单独计算即可。

由于值域不同，单纯的加法需要单独记录，处理时记录当前区间是否有求 popcount 即可。总时间复杂度 $O(q\log n\log V)$。

1. 模拟赛解析
2. 树形 DP
3. 状压 DP

P3574

给定一棵大小为 n 的树，每个点有一个权值 a_i。你需要从根节点 1 出发，dfs 遍历整棵树，最后回到根节点。设 t_i 为第一次遍历到点 i 时经过的边数，求出对于所有不同的 dfs 路径，$\max(a_1 + 2n - 2, \max_{i=2}^{n}\{a_i + t_i\})$ 的最小值。
$n \le 5 \times 10^5, 1 \le a_i \le 10^9$。

P3574 · 解

考虑设 $f(u)$ 为以 u 为根的子树内若以 u 为起点，$\max_{i\in Subtree(u)}\{a_i+t_i\}$ 的最小值。可以得到转移方程：

$$f(u)=\max_{v_i\ fa(v_i)=u}(f(v_i)+2\sum_{j<i} siz_{v_j}+1)$$

但是注意到若随意钦定枚举儿子的顺序则不一定能取到最优解。对于 u 的一个已有的儿子序列 $v_1,\dots,v_k$，考虑交换两个相邻项 v_i 和 v_j 是否能使 $f(u)$ 更小。可以发现交换 v_i,v_j 不会影响其他儿子对 $f(u)$ 的贡献，因此只需要比较 v_i,v_j 的贡献。

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

P3574 · 解

顺序为 v_i, v_j 时，两者贡献为 $f(v_i)+2S+1$ 和 $f(v_j)+2S+1+2siz(v_i)$；
顺序为 v_j, v_i 时，两者贡献为 $f(v_j)+2S+1$ 和 $f(v_i)+2S+1+2siz(v_j)$；其中 S 为在两者之前的儿子的子树大小之和。
可以注意到前者优于后者的充要条件是 $f(v_j)+2S+1+2siz(v_i) < f(v_i)+2S+1+2siz(v_j)$，化简得到 $f(v_i)-2siz(v_i) > f(v_j)-2siz(v_j)$。
不妨令 $g(x)=f(x)-2siz(x)$，则求解 $f(u)$ 时只需根据 g 对儿子降序排序后逐一合并即可。时间复杂度 $O(n\log n)$，唯一瓶颈是排序。

换根 DP

常见的树形 DP 通常维护的是子树内的 DP 信息。当需要求出以每个点为根时整棵树的 DP 状态时，一般就需要应用换根 DP。具体地，首先钦定一个点为根，通过一次 dfs 求出每个点子树内的 DP 信息，此时根节点的 DP 状态即为以此节点为根时整棵树的 DP 状态。接着再进行一次 dfs，自上而下地通过每个点的父节点状态更新自身的状态，求出以每个点为根时整棵树的 DP 状态。

P3478

给定一个 n 个点的树，求出一个结点，使得以这个结点为根时，所有结点的深度之和最大。若有多解输出任意一个即可。
$n \leq 10^6$。

P3478 · 解

首先可以钦定 1 为根节点，通过一次 dfs 求出每个节点的子树大小 siz 和以 1 为根时所有结点的深度之和 $f(1)$。
考虑第二次 dfs 过程中，将根节点从 u 转移至它的其中一个儿子 v。可以发现 v 子树内节点的深度都减少 1，其余节点的深度都增加 1，因此可以得到转移方程
$f(v) = f(u) - siz_v + (n - siz_v) = f(u) + n - 2siz_v$。时间复杂度 $O(n)$。

树形背包

模板题是 P2014。

设 $f(u,i)$ 为节点为 u 的子树内选 i 个点的最大收益，每次枚举一个儿子 v 与其进行合并，可以得到 $f'(u,i+j)=\max(f(u,i+j),f(u,i)+f(v,j))$。两边的枚举上界为 $\min(siz_u,m)$ 和 $\min(siz_v,m-i)$，更新完 $f(u)$ 后对 siz_u 进行更新。

若枚举上界不对 m 取 min，则算法的时间复杂度为 $O(n^2)$，从本质上可以认为每两个节点在其 lca 处发生 $O(1)$ 的信息合并，共发生了 $O(n^2)$ 次。

当加上取 min 时，时间复杂度为 $O(n\min(n,m))$。不妨设 $m<n$，子树大小不超过 m 的为小子树，子树大小超过 m 的为大子树。定义极大小子树为不被其他小子树包含的小子树，极小大子树为不包含其他大子树的大子树。

树形背包

考虑将单点合并为一个极大小子树 u 的复杂度，由上可知为 $O(siz_u^2)$。因此该操作总的复杂度为 $O(\sum siz_u^2)$，其中 $\sum siz_u = n$。可以证明当 $siz_u = m$ 时 $\sum siz_u^2$ 取到最大值 nm，因此该操作复杂度为 $O(nm)$。
考虑将极大小子树 v 合并为极大小子树 u 上，单次合并的复杂度为 $O(siz_v m)$。注意到每个极大小子树仅会被这样合并一次，因此该操作总的复杂度为 $O(m\sum siz_v) = O(nm)$。
考虑将所有极小大子树合并为整棵树，由于极小大子树互不包含，其个数不会超过 $\dfrac{n}{m}$ 个，而单次合并的复杂度为 $O(m^2)$，因此该操作总的复杂度为 $O(m\sum siz_v) = O(nm)$。
综上，树形背包的时间复杂度为 $O(nm)$。

P4516

给定一棵大小为 n 的树，在一个节点上设置监听器可以监听与它相邻的节点，但是不监听它本身。你想知道在 k 个节点上设置监听器能监听全部 n 个节点的方案数。对 10^9+7 取模。
$n \le 10^5, k \le \min(n, 100)$。

P4516 · 解

树形背包题。设 $f(u,i,0/1,0/1)$ 为以 u 为根的子树内放了 i 个监听器，u 点是否放监听器，u 点是否被监听时的方案数。注意除 u 点外子树内其他节点均已被监听。得到转移方程：

$$f'(u,i+j,0,0)=\sum f(u,i,0,0)\times f(v,j,0,1)$$

f'(u,i+j,1,0)=$\sum f(u,i,1,0)\times(f(v,j,0,0)+f(v,j,0,1))$
f'(u,i+j,0,1)=$\sum f(u,i,0,0)\times f(v,j,1,1)+\sum f(u,i,0,1)\times(f(v,j,0,1)+f(v,j,1,1))$
f'(u,i+j,1,1)=$\sum f(u,i,1,0)\times(f(v,j,1,1)+f(v,j,1,0))+\sum f(u,i,1,1)\times(f(v,j,0,0)+f(v,j,1,0)+f(v,j,0,1)+f(v,j,1,1))$

时间复杂度为 $O(nm)$。

1. 模拟赛解析
2. 树形 DP
3. 状压 DP

P2704

给定一个 $n \times m$ 的网格图 s，若 $s_{i,j}$ 为 P 则可以放置至多一支炮兵部队，若为 H 则不能放置。位于 (i,j) 的部队可以攻击到 $(i+k,j)$ 和 $(i,j+k)$，其中 $k \in [-2,2]$。求出在不会相互攻击到的情况下最多放置几支炮兵部队。

$n \leq 100, m \leq 10$。

P2704 · 解

考虑用二进制表示一行中哪些点放置了部队，设 $f(i, S_1, S_2)$ 为前 i 行中，第 i 行放置状态为 S_1，第 $(i-1)$ 行放置状态为 S_2 的方案数，转移时枚举第 $(i+1)$ 行的状态 S_0，判断是否会相撞以及能否放置即可。时间复杂度 $O(nS^3)$，其中 S 为一行内不会相撞的状态数。
虽然粗略地看 S_0 的量级是 $O(2^m)$ 的，但是当 $m \leq 10$ 时 S 的总合法状态数约为 60，可以通过此题。

P3959

给定一个 n 个点 m 条边的连通图，以一个点为根节点求得该图的一棵生成子图 T，则一条边的代价为其深度较浅的端点到根节点路径上的节点个数 k 乘上该边的权值 w。T 的代价为其包含的所有边的代价之和。求出 T 的代价的最小值。

$n \leq 12, m \leq 10^3$ $1 \leq w \leq 5 \times 10^5$。

P3959 · 解

朴素的想法是枚举当前并入根节点的集合 S，然后枚举 x 满足 $x \notin S$，计算 x 与 S 相连的最小代价。但是这样无法计算此边深度较浅的端点到根节点路径上的节点个数。

考虑从一个根节点出发，初始为一层，以类似 bfs 的方式每次向外扩展一层，只需额外维护一个当前扩展了几层即可。具体地，设 $f(i,S)$ 为并入根节点的集合是 S，已经扩展了 i 层时代价的最小值。预处理出每个点 x 与每个集合 S 的相连的最小边权 $g(x,S)$ 以及集合 S 向外扩展一步能到达的最大集合 $E(S)$，转移时枚举 $S' \in E(S) \backslash S$，转移方程即为

$f(i+1, S \cup S') = \min\{f(i,S) + i \cdot \sum_{x \in S'} g(x,S)\}$。

S' 的枚举和对应代价的复杂度均为 $O(3^n n)$，计算 $E(S)$ 的复杂度为 $O(2^n m)$，总时间复杂度为 $O(3^n n + 2^n m)$。

P2150

有 $(n-1)$ 个寿司，权值为 2 至 n。甲和乙想每个人选若干个寿司（可以不选）品尝，问甲选择的寿司的权值之积和乙选择的寿司的权值之积的方案数。对给定正整数 p 取模。

$2 \leq n \leq 500$。

P2150 · 解

注意到一个寿司的权值为若干质数之积，因此一个质数只能出现在至多一个人选择的寿司权值之积中。

考虑枚举甲的质数的集合 S_1，再在其补集中枚举乙的质数的集合 S_2，预处理出每个集合包含的寿司，即可求得甲只在 S_1 包含的寿司中选，乙只在 S_2 包含的寿司中选的方案数 $f(S_1,S_2)$。由于我们要计算恰好构成集合 S_1,S_2 的方案数 $g(S_1,S_2)$，需要再对 f 进行一次容斥。时间复杂度为 $O(3^{P(n)})$，其中 $P(n)$ 为大小不超过 n 的质数个数。

P2150 · 解

进一步观察可以发现，定义一个质数 p 为大质数当且仅当 $p^2 \geq n$，则一个寿司至多有一个大质数因子。因此我们可以考虑将寿司分类为有某个大质数以及没有大质数。设 $f(i, S_1, S_2)$ 为枚举了前 i 个大质数，甲的小质数集合为 S_1，乙的小质数集合为 S_2 时的方案数，在外层枚举 S_1, S_2，内部转移时枚举当前大质数被甲或乙或没有人选即可。最后仍需对方案数进行容斥。时间复杂度 $O(3^{P(\sqrt{n})} n)$。

关于容斥：一个很好的性质是若 $|S_1|, |S_2|$ 均相等，则其容斥系数也是相等的。因此可以按照 $|S_1|, |S_2|$ 分类并计算系数，此部分时间复杂度为 $O(P(\sqrt{n})^2)$。