NOIP 模拟题 Day2
Solution

Colin

1 排序

1.1 题意简述

给定一个栈和一个 n 的全排列作为入栈序列，试通过调整出栈次序，得到字典序最大的出栈序列。

1.2 解法一

暴力枚举每次是入栈还是出栈，即穷举所有可能的出栈序列，从而得到字典序最大的出栈序列。

时间复杂度：$O(C(n))$，其中 $C(n)$ 为卡特兰数列的第 n 项，期望得分 40 分。

1.3 解法二

使用各种神奇的贪心算法，求解字典序最大的出栈序列。由算法复杂度及正确性决定最后得分。

期望得分 $0-100$ 分。

1.4 解法三

依旧考虑贪心算法。要求字典序最大，故我们可以从 n 到 1 贪心地试探每个数能否加入出栈序列。假设我们已经枚举到了 i，若栈顶元素比 i 大，我们则可以弹出栈顶元素，将其加入出栈序列，直到栈顶元素 $\leq i$。

若 i 未在栈中，我们则将入栈序列中在 i 前面且未入栈的数入栈，并将 i 加入出栈序列；

若 i 在栈中且是栈顶元素，我们则将 i 弹出栈，并加入出栈序列；

若 i 已经在栈中且不是栈顶元素，则说明若将 i 加入出栈序列，之前必然要把比 i 小的数弹出栈，我们不这样做，继续枚举 $i-1$。

注意由于涉及大规模文件处理，此题需要使用输入输出优化。

时间复杂度为 $O(n)$，期望得分 100 分。

2 花花的森林

2.1 题意简述

见原题。

2.2 解法一

对于一棵树，我们可以通过 n 次遍历，求得树的直径。我们可以模拟每次删边操作，通过 $O(n^2)$ 的暴力遍历求得新联通块的直径，从而得到答案。

时间复杂度为 $O(n^3)$，期望得分 $40-60$ 分。

2.3 解法二

我们考虑倒着做，即最开始是一个包含了 n 棵只有一个点的树的森林，然后不断加边，最后得到一棵完整的树。这样我们只需能够快速求出两棵树合并后得到新树的直径即可解决这个问题。

设合并的两棵树是 T_a 和 T_b，合并后形成的新树是 T。记 T_a 的直径的两端点为 u_a 和 v_a，T_b 的直径的两端点为 u_b 和 v_b，则 T 的直径的两端点一定 $\in \{u_a, u_b, v_a, v_b\}$（由反证法不难证明）。这样我们只需维护每个块的直径及直径的两端点。在合并两个块时，我们只需暴力求出 4 个点对之间的路径长度，进行比较即可。

在这个过程中我们需要除以原来两块的直径。在模的意义下，除以一个数等价于乘以这个数的逆。

时间复杂度为 $O(n^2)$，期望得分 $60-80$ 分。

2.4 解法三

我们考虑改进解法二的做法，虽然有加边、删边操作，但事实上我们要求的点对之间的路径长度都是原树上的路径。故我们可以通过对原树进行倍增预处理，每次在 $O(logn)$ 的时间内求得点对间的最近公共祖先（`lca`），从而得到点对间的路径长度。

时间复杂度为 $O(nlogn)$，期望得分 100 分。

3 最短路

3.1 题意简述

见原题。

3.2 解法一

对于正权图的最短路问题，我们可以选择Dijkstra和SPFA等算法进行求解。

时间复杂度 $O(qnlogn)$，期望得分 60 分。

3.3 解法二

对于树上最短路问题，我们可以选择树上遍历或树上倍增等算法求解。

时间复杂度为 $O(nq)$ 或 $O(nlogn+qlogn)$，期望得分 $50-60$ 分。

3.4 解法三

对于只有一个环的情况，我们可以选择经典的环套树动态规划求解。

时间复杂度为 $O(nlogn+qlogn)$，期望得分 80 分。

3.5 解法四

这道题实际上为仙人掌图[1]上的点对最短路径求解问题，可以使用动态规划进行求解。

考虑树上倍增的思想，$dist(u,v)=dis[u]+dis[v]-2\times dis(lca(u,v))$，其中 $dis[u]$ 表示 u 到根的最短距离，$lca(u,v)$ 表示 u 和 v 的最近公共祖先，$dist(u,v)$ 表示 u 和 v 之间的最短路径。

对于仙人掌图我们任然使用这种思想，我们首先考虑如何求 $dis[u]$，即 u 到根的最短路径。对于非环上的路径可以类似树上动规。对于环上路径，我们设环中深度最小的节点为 r，则 $dis[u]=dis[r]+dist(u,r)$，其中 $dist(u,r)$ 有两种选择。我们可以通过Tarjan算法在遍历到 r 时，找到这个环，并对这个环进行处理，求出环上的 $dis[]$ 值。（实际上就是求出环上的一个分界点，左边的点都从一个方向走到 r，右边的点都从另一个方向走到 r。）这样我们便求得了 $dis[]$。

对于仙人掌图，当lca在环上时，环上的路径有两种走法。故我们考虑对仙人掌图重新构图。对于每个环，新设一个点，这个点连向环中深度最小的点 r，环中其他点都向这个点连边。这样在新图中求lca，我们就可以很容易地知道lca是否在一个环上，若在环上，我们还可以知道 u 和 v 分别是从哪两个点（设为 c_u 和 c_v）进入环的。

若lca不在环上，$dist(u,v)=dis[u]+dis[v]-2\times dis[lca(u,v)]$。

[1]每条边最多属于一个环的无向连通图。

若lca在环上，$dist(u,v) = dis[u]+dis[v]-dis[c_u]-dis[c_v]+dist(c_u,c_v)$。其中 $dist(c_u,c_v)$ 为 c_u 和 c_v 在环上的最短路径。通过在Tarjan时，预处理出环的大小，以及环上每个点沿一个方向走到 r 的距离，我们可以直接算出 $dist(c_u,c_v)$。

时间复杂度为 $O(nlogn+qlogn)$，期望得分 100 分。