城市改建

【题意简述】

n 个节点的树，要求删掉一条边后新加一条边，使得新树的直径最小。

【数据范围】

20%的数据满足：n≤30
60%的数据满足：n≤3000
100%的数据满足：1≤n≤300000

【算法分析】

算法 1：

枚举删去的边和新增的边，求出新树的直径，计算最小值。

求树的直径可以枚举一个点，再求距离它最远的点，这样做时间复杂度是 $O(n^2)$的。也可以通过两次 dfs(bfs)解决：先任选一个点，求出距离它最远的点，不难证明，这个点一定是某条直径的端点，那么，再从该点出发，求出距离它最远的点，即可得到直径，这样做时间复杂度是 O(n)的。

时间复杂度: $O(n^4)$或 $O(n^5)$

空间复杂度:$O(n)$

期望得分：20 分

算法 2：

枚举删去的边后，得到两棵子树 A、B，记子树 A 的直径为 d(A)，子树 B 的直径为 d(B)。设我们新加的边 e=(a,b)(a∈A、b∈B)，那么新树的直径为 max{d(A),d(B),A 中距 a 最远的点的距离+B 中距 b 最远的点的距离+1}。要使新树直径最小，显然，应选择 a 为子树 A 的中心节点，b 为子树 B 的中心节点。此时，新树的直径为$\max\{d(A),d(B),\left\lceil\frac{d(A)}{2}\right\rceil+\left\lceil\frac{d(B)}{2}\right\rceil+1\}$。

这样，我们便不需再枚举新加的边，只需分别求出两棵子树的直径即可。

时间复杂度:$O(n^2)$

空间复杂度:$O(n)$

期望得分：60 分

算法 3：

我们需要快速求出删去一条边后两棵子树的直径。

先任选一个点作为根节点。求以节点 i 为根的子树的直径可以用 dp 解决。设 f[i]表示以节点 i 为根的子树的直径，g[i]表示以节点 i 为根的子树中距离 i 最远的点的距离。则 f[i]=max{f[j]，g[j]+1，g[j]+1+g[k]+1|j、k 是 i 的孩子且 j≠k}，g[i]=max{g[j]+1|j 是 i 的孩子}。记录 i 的孩子中 g 的最大值和次大值即可得出 max{ g[j]+1+g[k]+1}。整个过程自底向上求解。

类似的，设 h[i]表示除去以节点 i 为根的子树后，剩下的子树的直径，p[i]表示在以节点 i 为根的子树外距离 i 最远的点的距离。这同样可以用 dp 解决，不过要稍微复杂一些。在计算完 f 和 g 后，自顶向下求解。显然，根节点的 h 值和 p 值为 0。设当前位于节点 i，h[i]和 p[i]已经求出，我们要计算出 i 的孩子的 h 值和 p 值。考虑孩子 j，h[j]=max{f[k]，h[i]，g[k]+1+g[l]+1，g[k]+1+p[i] | k、l 是 i 的孩子且 k≠j、l≠j、k≠l }，p[j]=max{g[k]+2,p[i]+1 | k 是 i 的孩子且 k≠j }。若是每次求 h[j]、p[j]都扫描一遍 i 的其他孩子，时间复杂度可能退化为 $O(n^2)$。事实上，这是非常没有必要的。我们可以依次求解 i 的每个孩子，用 pre[j]记录前 j 个孩子的信息(包括 f 的最大值、g 的最大值和次大值)，用 suf[j]记录第 j 个孩子至最后一个孩子的信息，那么，求解第 j 个孩子时，只需整合 pre[j-1]和 suf[j+1]的信息再加上与 i 有关的计算即可。这样就能做到线性的时间复杂度了。

最后 $\max\{f[i], h[i], \left\lceil \frac{f[i]}{2} \right\rceil + \left\lceil \frac{h[i]}{2} \right\rceil + 1\}$ 就是删去边(i,fa[i])后新树的最小直径。

由于本题还要求输出方案，上述算法还需记录是从哪个点更新而来的。实现起来稍有些繁琐，要耐心、细心完成。

时间复杂度:*O(n)*

空间复杂度:*O(n)*

期望得分：100 分