2025/11/29 20:42 1/2 树链剖分

# 树链剖分

<del>用输入法打这个词永远联想不出来</del> 用于处理有根树上路径问题

#### 原理

对于有根树tree 不妨定义它的树根为 1 对于每棵子树,定义它的 size[i]=\$\Sigma\_{j\in son[i]}\$ size[j] +1

我们定义重儿子 hson[i] ,使得 size[hson[i]]=size(deep[son[i]])即重儿子有最大的子树的儿子显然,从任意一个点不断选择重儿子向下,最终可以到达叶,这条路径称为重链从任意点到它的轻儿子的边称为轻链

记录 top[i] 为: i 只经过重链向上可以到达的最高的点记录dfs序 dfn[i], 这个dfs序先遍历重儿子。那么,每一条重链在dfs序中是连续的

那么,对于路径(i,j),将它拆为(i,lca(i,j),j)lca(i,j)必然是i的祖先(显然)路径(i,lca(i,j))必然包含若干条重链和相同数量(\$\pm 1\$)的轻链由于每条重链在dfs序中是连续的一段,所以可以用线段树等结构区间修改轻链上的点单点修改就能够维护树上路径了

### 复杂度证明

只证明从i到根之多需要经过  $SO(\log_2 n)$  条轻链 对于边 (i,j) j 是 i 的轻儿子,有 Size[i] >= Size[j]\*2 所以经过一条轻链,子树大小至少翻倍。得证

#### 实现

dfs两遍。 dfs1求出 father□size□hson 以及其他信息 (如果需要,比如depth) dfs2 求出top□dfn

# 一点细节

#### 求lca

树上路径必然需要lca 求 lca(i,j): 不妨令 depth[i]>depth[j](不满足就交换) 调整 i=father[top[i]],(此时如果不满足depth[i]>depth[j]则再交换)  $\frac{\text{update:}}{2020/05/13} \\ 2020-2021: \text{teams:} \\ \text{no\_morning\_training:} \\ \text{fayuanyu:} \\ \text{pou https://wiki.cvbbacm.com/doku.php?id=2020-2021:} \\ \text{teams:} \\ \text{no\_morning\_training:} \\ \text{fayuanyu:} \\ \text{pou\&rev=1589362930} \\ \text{teams:} \\ \text{fayuanyu:} \\ \text{pou\&rev=1589362930} \\ \text{teams:} \\ \text{fayuanyu:} \\ \text{fayua$ 

直到top[i]==top[j]。 这时 i, j 中 depth较小者即为 lca(i, j)

正确性证明略去

# 下期预告

树剖要求这颗树是有根并且静态的如果树需要换根,加点,删点,请使用 ||ct 如果需要在此基础上求子树信息(),请使用 |toptree

From

https://wiki.cvbbacm.com/ - CVBB ACM Team

Permanent link

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:no\_morning\_training:fayuanyu:pou&rev=1589362930

Last update: 2020/05/13 17:42



https://wiki.cvbbacm.com/ Printed on 2025/11/29 20:42