树链剖分

~~用输入法打这个词永远联想不出来~~
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(± 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 $O(\log_2 n)$ 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)

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 |lct
如果需要在此基础上求子树信息()， 请使用 |toptree