点分治

Centroid Decomposition

Idea：树上统计问题中，可分治地统计——先统计经过树根的路径，再统计各个的子树的答案。每次选择重心作为树的根节点，以保证时间复杂度。根据题目的不同，统计的方法也不同（即 `calc` 函数不同），而特别地，一些统计问题需要容斥或减去重复统计的答案。

Complexity：$O(n \lg n)$

ATT：每一次 `getRoot` 前记得初始化。

Code：

```
int root, sum, mxson[N], size[N];
bool vis[N];
void getRoot(int x, int f){
    mxson[x] = 0, size[x] = 1;
    for(int i = head[x]; i; i = edge[i].nxt){
        if(edge[i].to == f || vis[edge[i].to])  continue;
        getRoot(edge[i].to, x);
        size[x] += size[edge[i].to];
        mxson[x] = max(mxson[x], size[edge[i].to]);
    }
    mxson[x] = max(mxson[x], sum - size[x]);
    if(mxson[root] > mxson[x])  root = x;
}

void calc(int x){
    ...; //根据题目写相应的统计函数
}

void solve(int x){
    calc(x, 1); // 1 是为容斥方便，依题而定
    vis[x] = 1;
    for(int i = head[x]; i; i = edge[i].nxt){
        if(vis[edge[i].to]) continue;
        calc(edge[i].to, -1); // 容斥原理，依题确定统计是否需要容斥
        root = 0, sum = size[edge[i].to], mxson[0] = INF; // 初始化
        getRoot(edge[i].to, 0);
        solve(root);
    }
}

int main(){
    ...;
    root = 0, sum = n, mxson[0] = INF; // 每一次 getRoot 记得初始化
    getRoot(1, 0);
    solve(root);
    ...;
}
```