树上启发式合并

DSU on Tree

Idea：如果暴力做，我们会整体做一次 dfs 以遍历每个点，然后对遍历的每个点 dfs 它的子树，计算答案，把答案记录抹去，算下一个点。这样做是 $O(n^2)$ 的。为优化，我们遍历到一个点时，先遍历它的轻儿子，计算轻儿子的答案并抹去记录，最后计算它的重儿子的答案且**不抹去**记录，如此在计算它本身的答案时，只需要暴力算出轻儿子的答案，合并上原本就存在的重儿子答案即可。

代码中的 `dsu()` 函数对应暴力算法的整体 dfs，`getData()` 函数对应暴力算法的“对遍历的每个点 dfs 计算子树答案”。

Complexity：$O(n \lg n)$

Code：

```cpp
int fa[N], sz[N], son[N];
void dfs(int x, int f){
    fa[x] = f, sz[x] = 1, son[x] = 0;
    for(int i = head[x]; i; i = edge[i].nxt){
        if(edge[i].to == f) continue;
        dfs(edge[i].to, x);
        sz[x] += sz[edge[i].to];
        if(!son[x] || sz[edge[i].to] > sz[son[x]])
            son[x] = edge[i].to;
    }
}

LL ans[N], mx, sum, cnt[N]; // GLOBAL variants to store the answer
int mark; // mark the heavy son which needs to be ignored
void getData(int x, int val){ // get data with brute-force

    cnt[c[x]] += val;
    if(mx < cnt[c[x]])  mx = cnt[c[x]], sum = c[x];
    else if(mx == cnt[c[x]])    sum += c[x];

    for(int i = head[x]; i; i = edge[i].nxt){
        if(edge[i].to == fa[x]) continue;
        if(edge[i].to == mark)  continue; // ignore the marked subtree
        getData(edge[i].to, val);
    }
}
void dsu(int x, bool opt){ // opt == true: answer needs to be erased
    for(int i = head[x]; i; i = edge[i].nxt){
        if(edge[i].to == fa[x] || edge[i].to == son[x]) continue;
        dsu(edge[i].to, true); // solve for light sons first
    }
    if(son[x])  dsu(son[x], false), mark = son[x]; // solve for heavy son
    // now the global variants have already stored heavy son's answer
    getData(x, 1);
    mark = 0;

    // now the global variants store the answer for vertex x
    ans[x] = sum;

    if(opt){ // erase the answer
        getData(x, -1);
        mx = 0, sum = 0;
    }
}

int main(){
    scanf("%d", &n);
```

```
    for(int i = 1; i <= n; i++)
        scanf("%d", &c[i]);
    for(int i = 1; i < n; i++){
        int u, v; scanf("%d%d", &u, &v);
        addEdge(u, v), addEdge(v, u);
    }
    dfs(1, 0);
    dsu(1, true);
    for(int i = 1; i <= n; i++)
        printf("%lld ", ans[i]);
    return 0;
}
```