树链剖分

支持操作

在子树上或者是两点路径上的所有节点的求和、修改、查询等操作。

具体操作大致为：

（1）dfs进行树链剖分处理节点信息；

（2）根据dfs序把树拉直然后套数据结构。

dfs序

把树搞成了“连续的”。

一个节点的子树上的节点的时间戳，一定大于这个节点的时间戳且连续。

某些链上的时间戳也是连续的。

因为dfs序的性质——重链上的时间戳是连续的。

树链剖分

重链剖分——常用

$O(logn)$

将每个点子树大小最大的儿子标记为重儿子。

长链剖分——不常用

$O(\sqrt{n})$

长链剖分十分类似于轻重链剖分，但是我们稍加修改，将每次选择子树大小最大的儿子作为重儿子变成了选择子树深度最大的那个儿子作为重儿子。然后将所有点和它的重儿子之间的边认为是重边，如果我们把他们在树中全部加粗，那么原树就被分割成了若干条链。

性质

（1）所有链的长度和是O(n)级别的。

（2）任意一点x的k级祖先y所在的长链长度大于等于k

（3）任意一点到根节点的重链数量不超过\sqrt n

应用

（1）O(nlogn)预处理，O(1)算k次祖先。

对于每一条长链的顶端x,记长链长度为L，预处理x沿长链从上往下的每一个点，以及x的L个祖先。要求点u的k次祖先，取$r = 2^p$(p+1是k二进制的最高位)，r>k/2,u的r次祖先由预处理得到，记为y。由性质2，y的k-r次祖先可直接求得。

（2）O(n)统计每个子树中以深度为下标的可合并信息。
用指针维护动规数组，在重儿子传递给父亲时将指针左移/右移一位。对于轻儿子，暴力统计信息。

实链剖分（搞LCT）

重儿子：一个节点的所有儿子中，大小最大的儿子（只有1个，如果相等则随便选）

轻儿子：一个节点除了重儿子之外的儿子都是轻儿子

重链：从一个轻儿子开始（根节点也是轻儿子），一路向重儿子走，连出的链叫做重链。除根节点外的任何一个节点的父亲节点都一定在一条重链上。

轻链：除了重链全是轻链

剖分好再dfs一遍标出dfs序即可实现上述四种操作。

树链剖分与线段树

树链剖分和线段树可以结合使用，来解决树上路径修改、查询的问题。

LuoguP4315 月下“毛景树”

在一棵带边权树上支持以下三种操作:

- Change k w : 将第k条边的边权改为w。
- Cover u v w : 将结点u和结点v之间的边权都变为w。
- Add u v w : 将结点u和结点v之间的边权都增加w

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int N=2e5+7;
const int mod=1e9+7;

struct E{
    int v,w,n;
}e[N<<2];

int head[N],bk[N<<1],cnt=0;
inline void addedge(int u,int v,int w){
    e[++cnt]=(E){v,w,head[u]};head[u]=cnt;
    e[++cnt]=(E){u,w,head[v]};head[v]=cnt;
}

int sz[N],fa[N],son[N],val[N],dep[N],dfn[N],idx,idfn[N],bel[N];
void dfs1(int x){
    sz[x]=1;
    for(int i=head[x];i;i=e[i].n){
        int to=e[i].v;
        if(to==fa[x]) continue;
        fa[to]=x;
        val[to]=e[i].w;
        dep[to]=dep[x]+1;
        dfs1(to);
        sz[x]+=sz[to];
        if(sz[son[x]]<sz[to]) son[x]=to;
    }
}

void dfs2(int x,int tp){
    dfn[x]=++idx;
    idfn[idx]=x;
    bel[x]=tp;
    if(son[x]) dfs2(son[x],tp);
    for(int i=head[x];i;i=e[i].n){
        int to=e[i].v;
        if(to==son[x]||to==fa[x]) continue;
        dfs2(to,to);
    }
}

int tr[N<<2],lazy[N<<2],tg[N<<2];
#define MID int mid=l+r>>1
#define ls p<<1
#define rs p<<1|1
void build(int p,int l,int r){
    tg[p]=-1;
    if(l==r){
```

```
        tr[p]=val[idfn[l]];
        return;
    }
    MID;
    build(ls,l,mid);
    build(rs,mid+1,r);
    tr[p]=max(tr[ls],tr[rs]);
}

void pushdown(int p,int l,int r){
    if(tg[p]>=0){
        lazy[ls]=lazy[rs]=0;
        tg[ls]=tg[p];tg[rs]=tg[p];
        tr[ls]=tg[p];tr[rs]=tg[p];
        tg[p]=-1;
    }
    if(lazy[p]){
        lazy[ls]+=lazy[p];
        lazy[rs]+=lazy[p];
        tr[ls]+=lazy[p];
        tr[rs]+=lazy[p];
        lazy[p]=0;
    }
}

void cover(int p,int l,int r,int ql,int qr,int w){
    if(ql<=l&&r<=qr){
        tr[p]=w;
        tg[p]=w;
        lazy[p]=0;
        return;
    }
    MID;
    pushdown(p,l,r);
    if(ql<=mid) cover(ls,l,mid,ql,qr,w);
    if(qr>mid) cover(rs,mid+1,r,ql,qr,w);
    tr[p]=max(tr[ls],tr[rs]);
}

void add(int p,int l,int r,int ql,int qr,int w){
    if(ql<=l&&r<=qr){
        tr[p]+=w;
        lazy[p]+=w;
        return;
    }
    MID;
    pushdown(p,l,r);
    if(ql<=mid) add(ls,l,mid,ql,qr,w);
    if(qr>mid) add(rs,mid+1,r,ql,qr,w);
    tr[p]=max(tr[ls],tr[rs]);
}

int query(int p,int l,int r,int ql,int qr){
    if(ql<=l&&r<=qr) return tr[p];
    MID;int res=0;
    pushdown(p,l,r);
    if(ql<=mid) res=max(res,query(ls,l,mid,ql,qr));
    if(qr>mid) res=max(res,query(rs,mid+1,r,ql,qr));
```

```
    return res;
}

int n,u,v,w,rt;
string str;

signed main(){
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    //freopen("in.cpp","r",stdin);
    //freopen("out.cpp","w",stout);
    cin>>n;
    for(int i=1;i<n;i++){
        cin>>u>>v>>w;
        addedge(u,v,w);
        bk[i]=v;
    }
    for(int i=1;i<=n;i++) if(bk[i]) rt=i;
    dfs1(rt);
    dfs2(rt,rt);
    build(1,1,n);
    while(cin>>str){
        if(str=="Stop") break;
        if(str=="Max"){
            cin>>u>>v;
            int res=0;
            while(bel[u]!=bel[v]){
                if(dep[bel[u]]<dep[bel[v]]) swap(u,v);
                res=max(res,query(1,1,n,dfn[bel[u]],dfn[u]));
                u=fa[bel[u]];
            }
            if(dep[u]<dep[v]) swap(u,v);
            if(u!=v) res=max(res,query(1,1,n,dfn[v]+1,dfn[u]));
            cout<<res<<"\n";
        }
        if(str=="Add"){
            cin>>u>>v>>w;
            while(bel[u]!=bel[v]){
                if(dep[bel[u]]<dep[bel[v]]) swap(u,v);
                add(1,1,n,dfn[bel[u]],dfn[u],w);
                u=fa[bel[u]];
            }
            if(dep[u]<dep[v]) swap(u,v);
            if(u!=v) add(1,1,n,dfn[v]+1,dfn[u],w);
        }
        if(str=="Change"){
            cin>>u>>v;
            u=dep[e[u*2-1].v]<dep[e[u<<1].v]?e[u<<1].v:e[u*2-1].v;
            cover(1,1,n,dfn[u],dfn[u],v);
        }
        if(str=="Cover"){
            cin>>u>>v>>w;
            while(bel[u]!=bel[v]){
                if(dep[bel[u]]<dep[bel[v]]) swap(u,v);
                cover(1,1,n,dfn[bel[u]],dfn[u],w);
                u=fa[bel[u]];
            }
            if(dep[u]<dep[v]) swap(u,v);
```

```
            if(u!=v) cover(1,1,n,dfn[v]+1,dfn[u],w);
        }
    }
    return 0;
}
```

luoguP3833 [SHOI2012]魔法树

给定一棵带点权的树，支持以下操作：

- A u v d，将u到v路径上的所有结点权值加上d
- Q u，询问以u为根的子树中的总果子树。

```
#include<bits/stdc++.h>
#define inf 0x3f3f3f3f
#define int long long
using namespace std;
const int N=2e5+7;
const int mod=1e9+7;

//int read(){   int x=0,f=1;char ch=getchar();while(ch<'0'||ch>'9'){if(ch=='-')
f=f*-1;ch=getchar();}while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}return
x*f;}
vector<int>G[N];
inline void addedge(int u,int v){
    G[u].push_back(v);
    G[v].push_back(u);
}

int sz[N],dep[N],fa[N],son[N],bel[N],dfn[N],idx;
inline void dfs1(int x){
    sz[x]=1;
    for(int i=0;i<G[x].size();i++){
        int to=G[x][i];
        if(to==fa[x]) continue;
        fa[to]=x;
        dep[to]=dep[x]+1;
        dfs1(to);
        sz[x]+=sz[to];
        if(sz[son[x]]<sz[to]) son[x]=to;
    }
}

inline void dfs2(int x,int tp){
    dfn[x]=++idx;
    bel[x]=tp;
    if(son[x]) dfs2(son[x],tp);
    for(int i=0;i<G[x].size();i++){
        int to=G[x][i];
        if(to==son[x]||to==fa[x]) continue;
        dfs2(to,to);
    }
}

#define MID int mid=l+r>>1;
#define ls p<<1
```

```
#define rs p<<1|1
int tr[N<<2],lazy[N<<2];

void pushdown(int p,int l,int r){
    if(lazy[p]){
        MID;
        lazy[ls]+=lazy[p];
        lazy[rs]+=lazy[p];
        tr[ls]+=(mid-l+1)*lazy[p];
        tr[rs]+=(r-mid)*lazy[p];
        lazy[p]=0;
    }
}

void update(int p,int l,int r,int ql,int qr,int val){
    if(ql<=l&&r<=qr){
        lazy[p]+=val;
        tr[p]+=(r-l+1)*val;
        return;
    }
    MID;
    pushdown(p,l,r);
    if(ql<=mid) update(ls,l,mid,ql,qr,val);
    if(qr>mid) update(rs,mid+1,r,ql,qr,val);
    tr[p]=tr[ls]+tr[rs];
}

int query(int p,int l,int r,int ql,int qr){
    if(ql<=l&&r<=qr) return tr[p];
    MID;int res=0;
    pushdown(p,l,r);
    if(ql<=mid) res+=query(ls,l,mid,ql,qr);
    if(qr>mid) res+=query(rs,mid+1,r,ql,qr);
    return res;
}

int n,u,v,q,d;
char op;

signed main(){
//  ios::sync_with_stdio(0);//  cin.tie(0);cout.tie(0);
//  freopen("in.cpp","r",stdin);freopen("out.cpp","w",stout);
    cin>>n;
    for(int i=1;i<n;i++){
        cin>>u>>v;
        addedge(u+1,v+1);
    }
    dfs1(1);
    dfs2(1,1);
    cin>>q;
    while(q--){
        cin>>op;
        if(op=='A'){
            cin>>u>>v>>d;
            u+=1;v+=1;
            while(bel[u]!=bel[v]){
                if(dep[bel[u]]<dep[bel[v]]) swap(u,v);
                update(1,1,n,dfn[bel[u]],dfn[u],d);
```

```
                u=fa[bel[u]];
            }
            if(dep[u]<dep[v]) swap(u,v);
            update(1,1,n,dfn[v],dfn[u],d);
        }else{
            cin>>u;
            u+=1;
            cout<<query(1,1,n,dfn[u],dfn[u]+sz[u]-1)<<"\n";
        }
    }
    return 0;
}
```

SDOI2014]旅行

在树上支持以下操作：

- `C x c`：修改结点 x 的颜色为c。
- `CW x w`：修改结点 x 的权值为w。
- `QS x y`：询问从 x 结点到 y 结点路径上颜色与 y 相同点的权值和。
- `QM x y`：询问从 x 结点到 y 结点路径上颜色与 y 相同点的权值和。

```
#include<bits/stdc++.h>
#define max(a,b) (a>b?a:b)
using namespace std;
typedef long long ll;
const int N=1e5+7;
int read(){ int x=0;char
ch=getchar();while(ch<'0'||ch>'9')ch=getchar();while(ch>='0'&&ch<='9')
{x=x*10+ch-'0';ch=getchar();}return x;}

vector<int>G[N];
inline void addedge(int u,int v){
    G[u].push_back(v);
    G[v].push_back(u);
}

int sz[N],fa[N],son[N],dep[N],dfn[N],idfn[N],bel[N],idx;
inline void dfs1(int x){
    sz[x]=1;
    for(int i=0;i<G[x].size();i++){
        int to=G[x][i];
        if(to==fa[x]) continue;
        fa[to]=x;
        dep[to]=dep[x]+1;
        dfs1(to);
        sz[x]+=sz[to];
        if(sz[son[x]]<sz[to]) son[x]=to;
    }
}

inline void dfs2(int x,int tp){
    dfn[x]=++idx;
    idfn[idx]=x;
    bel[x]=tp;
```

```
    if(son[x]) dfs2(son[x],tp);
    for(int i=0;i<G[x].size();i++){
        int to=G[x][i];
        if(to==son[x]||to==fa[x]) continue;
        dfs2(to,to);
    }
}

#define ls tr[p].lson
#define rs tr[p].rson
#define MID int mid=l+r>>1

int rt[N],SZ;
struct node{
    int lson,rson,mx;
    ll sum;
}tr[N*24];

inline void pushup(int p){
    tr[p].sum=tr[ls].sum+tr[rs].sum;
    tr[p].mx=max(tr[ls].mx,tr[rs].mx);
}

inline void build(int &p,int l,int r,int pos,int val){
    if(!p) p=++SZ;
    if(l==r){
        tr[p].mx=tr[p].sum=val;
        return;
    }
    MID;
    if(pos<=mid) build(ls,l,mid,pos,val);
    else build(rs,mid+1,r,pos,val);
    pushup(p);
}

inline void del(int &p,int l,int r,int pos){
    if(!p) return; //剪枝
    if(l==r){
        tr[p].mx=tr[p].sum=0;
        return;
    }
    MID;
    if(pos<=mid) del(ls,l,mid,pos);
    else del(rs,mid+1,r,pos);
    pushup(p);
}

inline ll qs(int &p,int l,int r,int ql,int qr){
    if(!p) return 0;
    if(ql<=l&&r<=qr) return tr[p].sum;
    MID;ll res=0;
    if(ql<=mid) res=qs(ls,l,mid,ql,qr);
    if(qr>mid) res+=qs(rs,mid+1,r,ql,qr);
    return res;
}

inline ll qm(int &p,int l,int r,int ql,int qr){
    if(!p) return 0;
```

```
    if(ql<=l&&r<=qr) return tr[p].mx;
    MID;ll res=0;
    if(ql<=mid) res=qm(ls,l,mid,ql,qr);
    if(qr>mid) res=max(res,qm(rs,mid+1,r,ql,qr));
    return res;
}

int n,m,x,y,tmp,val[N],col[N];
ll ans;
char op[10];

signed main(){
    n=read();m=read();
    for(int i=1;i<=n;i++) val[i]=read(),col[i]=read();
    for(int i=1;i<n;i++){
        x=read(),y=read();
        addedge(x,y);
    }
    dfs1(1);
    dfs2(1,1);
    for(int i=1;i<=n;i++) build(rt[col[idfn[i]]],1,n,i,val[idfn[i]]);
    for(int i=1;i<=m;i++){
        scanf("%s",op);
        x=read();y=read();
        if(op[1]=='C'){
            del(rt[col[x]],1,n,dfn[x]);
            col[x]=y;
            build(rt[col[x]],1,n,dfn[x],val[x]);
        }else if(op[1]=='W'){
            del(rt[col[x]],1,n,dfn[x]);
            val[x]=y;
            build(rt[col[x]],1,n,dfn[x],val[x]);
        }else if(op[1]=='S'){
            ans=0;
            tmp=col[x];
            while(bel[x]!=bel[y]){
                if(dep[bel[x]]<dep[bel[y]]) swap(x,y);
                ans+=qs(rt[tmp],1,n,dfn[bel[x]],dfn[x]);
                x=fa[bel[x]];
            }
            if(dep[x]<dep[y]) swap(x,y);
            ans+=qs(rt[tmp],1,n,dfn[y],dfn[x]);
            printf("%lld\n",ans);
        }else{
            ans=0;
            tmp=col[x];
            while(bel[x]!=bel[y]){
                if(dep[bel[x]]<dep[bel[y]]) swap(x,y);
                ans=max(ans,qm(rt[tmp],1,n,dfn[bel[x]],dfn[x]));
                x=fa[bel[x]];
            }
            if(dep[x]<dep[y]) swap(x,y);
            ans=max(ans,qm(rt[tmp],1,n,dfn[y],dfn[x]));
            printf("%lld\n",ans);
        }
    }
    return 0;
}
```

[Vani有约会]雨天的尾巴 /【模板】线段树合并

vp时写了一个树链剖分+权值线段树过了

```
#include<bits/stdc++.h>
#define pb emplace_back
#define pii pair<int,int>
#define mk make_pair
#define F first
#define S second
using namespace std;
const int N=1e5+7;
inline int read(){
    int x=0;char ch=getchar();
    while(!isdigit(ch)) ch=getchar();
    while(isdigit(ch)) x=x*10+ch-'0',ch=getchar();
    return x;
}
int n,m,ans[N];
vector<int>G[N];
int sz[N],fa[N],son[N],dep[N];
int dfn[N],idfn[N],idx=0,bel[N],cnt[N];
unordered_map<int,int>mp[N];
#define mid ((l+r)>>1)
#define inf (1e9+7)
int tot=0,rt;
struct Tr{
    int ls,rs,mx,mxid;
}tr[N<<5];
void update(int &p,int l,int r,int pos,int val){
    if(!p) p=++tot;
    if(l==r){
        tr[p].mx+=val;
        tr[p].mxid=pos;
        return;
    }
    tr[p].mx=tr[p].mxid=0;
    if(pos<=mid) update(tr[p].ls,l,mid,pos,val);
    else update(tr[p].rs,mid+1,r,pos,val);
    if(tr[p].ls&&tr[tr[p].ls].mx>tr[p].mx){
        tr[p].mx=tr[tr[p].ls].mx;
        tr[p].mxid=tr[tr[p].ls].mxid;
    }
    if(tr[p].rs&&tr[tr[p].rs].mx>tr[p].mx){
        tr[p].mx=tr[tr[p].rs].mx;
        tr[p].mxid=tr[tr[p].rs].mxid;
    }
}

inline void dfs1(int x,int f){
    sz[x]=1;
    for(auto to:G[x]){
        if(to==f) continue;
        fa[to]=x;dep[to]=dep[x]+1;
        dfs1(to,x);
```

```
        sz[x]+=sz[to];
        if(sz[son[x]]<sz[to]) son[x]=to;
    }
}

void dfs2(int x,int tp){
    dfn[x]=++idx;
    idfn[idx]=x;
    bel[x]=tp;
    if(son[x]) dfs2(son[x],tp);
    for(auto to:G[x]){
        if(to==son[x]||to==fa[x]) continue;
        dfs2(to,to);
    }
}

void modify(int x,int y,int z){
    mp[x][z]++;
    mp[y+1][z]--;  //使用map来差分颜色
}

signed main(){
    n=read();m=read();
    for(int i=1,u,v;i<n;++i){
        u=read();v=read();
        G[u].pb(v);G[v].pb(u);
    }
    dfs1(1,0);
    dfs2(1,1);
    for(int i=1,u,v,w;i<=m;++i){
        u=read();v=read();w=read();
        while(bel[u]!=bel[v]){
            if(dep[bel[u]]<dep[bel[v]]) swap(u,v);
            modify(dfn[bel[u]],dfn[u],w);
            u=fa[bel[u]];
        }
        if(dep[u]<dep[v]) swap(u,v);
        modify(dfn[v],dfn[u],w);
    }
    for(int i=1;i<=n;++i){
        for(auto it=mp[i].begin();it!=mp[i].end();++it) update(rt,1,inf,(it->F),
(it->S));
        ans[idfn[i]]=(tr[1].mxid);cnt[idfn[i]]=(tr[1].mx);
    }
    for(int i=1;i<=n;++i){
        if(!cnt[i]) puts("0");
        else printf("%d\n",ans[i]);
    }
    return 0;
}
```

线段树合并的做法

```
#include<cstdio>
#include<algorithm>
#include<vector>
using namespace std;const int N=1e5+10;typedef long long ll;
```

```
int n;int m;int v[2*N];int x[2*N];int ct;int al[N];int fa[N][22];int dep[N];int
ans[N];
struct data{//存最大值的结构体
    int u;int cnt;
    friend bool operator <(data a,data b){return (a.cnt==b.cnt)?
a.u>b.u:a.cnt<b.cnt;}
    friend data operator +(data a,data b){return (data){a.u,a.cnt+b.cnt};}
};vector <data> ve[N];
inline void add(int u,int V){v[++ct]=V;x[ct]=al[u];al[u]=ct;}
struct linetree{//动态开点线段树合并
    int s[40*N][2];data v[40*N];int ct;
    inline void ih(){ct=n;}
    inline void ins(int p,int l,int r,data va){//插入
        if(r-l==1){v[p]=va+v[p];return;}int mid=(l+r)/2;
        if(va.u<=mid)ins(s[p][0]=s[p][0]?s[p][0]:++ct,l,mid,va);
        else ins(s[p][1]=s[p][1]?s[p][1]:++ct,mid,r,va);v[p]=max(v[s[p]
[0]],v[s[p][1]]);
    }
    inline void mg(int p1,int p2,int l,int r){//合并
        if(r-l==1){v[p1]=v[p1]+v[p2];return;}int mid=(l+r)/2;//分情况讨论下
        if(s[p1][0]&&s[p2][0])mg(s[p1][0],s[p2][0],l,mid);else if(s[p2][0])s[p1]
[0]=s[p2][0];
        if(s[p1][1]&&s[p2][1])mg(s[p1][1],s[p2][1],mid,r);else if(s[p2][1])s[p1]
[1]=s[p2][1];
        v[p1]=max(v[s[p1][0]],v[s[p1][1]]);
    }
}lt;
inline void dfs1(int u){//处理lca
    for(int i=0;fa[u][i];i++)fa[u][i+1]=fa[fa[u][i]][i];dep[u]=dep[fa[u][0]]+1;
    for(int i=al[u];i;i=x[i])if(v[i]!=fa[u][0])fa[v[i]][0]=u,dfs1(v[i]);
}
inline int lca(int u,int v){//求lca
    if(dep[u]<dep[v])swap(u,v);for(int d=dep[u]-
dep[v],i=0;d;d>>=1,i++)if(d&1)u=fa[u][i];
    if(u==v)return u;for(int i=18;i>=0;i--)if(fa[u][i]!=fa[v][i])u=fa[u]
[i],v=fa[v][i];
    return fa[u][0];
}
inline void dfs2(int u){//线段树合并
    for(int i=al[u];i;i=x[i])if(v[i]!=fa[u][0])dfs2(v[i]),lt.mg(u,v[i],0,1e5);
    vector <data> :: iterator it;
    for(it=ve[u].begin();it!=ve[u].end();++it){lt.ins(u,0,1e5,*it);}
    ans[u]=lt.v[u].u;
}
int main(){
    scanf("%d%d",&n,&m);lt.ih();
    for(int i=1,u,v;i<n;i++){scanf("%d%d",&u,&v);add(u,v),add(v,u);}dfs1(1);
    for(int i=1,u,v,va;i<=m;i++){//拆成4个操作
        scanf("%d%d%d",&u,&v,&va);int lc=lca(u,v);
        ve[u].push_back((data){va,1});ve[v].push_back((data){va,1});
        ve[lc].push_back((data){va,-1});ve[fa[lc][0]].push_back((data){va,-1});
    }dfs2(1);for(int i=1;i<=n;i++)printf("%d\n",ans[i]);return 0;//拜拜程序~
}
```

参考资料

https://www.bilibili.com/video/BV1Qt411u77f?from=search&seid=6685906805699952409

https://www.bilibili.com/video/BV1RT4y1L7Sb