

树链剖分

是什么

待开发。大概介绍一下轻重链什么的

怎么用

普通用法

[BZOJ 2243][SDOI2011]染色 | 线段树

code:

```
#include<iostream>
#include<cstdio>
#include<cstring>
#define MAXN 100005
#define RG register
using namespace std;
int n,m,head[MAXN],cnt=0,ans,col[MAXN];
int
sz=0,pos[MAXN],top[MAXN],dep[MAXN],son[MAXN],father[MAXN],siz[MAXN],inv[MA
N];
inline int read()
{
    int f=1,x=0;char c=getchar();
    while(c<'0' || x>'9'){if(c=='-')f=-1;c=getchar();}
    while(c>='0'&&c<='9'){x=x*10+c-'0';c=getchar();}
    return f*x;
}
struct Node1{
    int next,to;
}Edges[MAXN*2];
inline void addedge(int u,int v)
{
    Edges[++cnt].next=head[u];
    head[u]=cnt;
    Edges[cnt].to=v;
}
inline void dfs1(int u)
{
    siz[u]=1;
    for(int i=head[u];~i;i=Edges[i].next)
    {
```

```
        int v=Edges[i].to;
        if(v==father[u])continue;
        father[v]=u;dep[v]=dep[u]+1;
        dfs1(v);siz[u]+=siz[v];
        if(siz[v]>siz[son[u]])son[u]=v;
    }
}
inline void dfs2(int u,int t)
{
    sz++,pos[u]=sz;
    inv[sz]=u,top[u]=t;
    if(son[u])dfs2(son[u],t);
    for(int i=head[u];~i;i=Edges[i].next)
    {
        int v=Edges[i].to;
        if(v==father[u]||v==son[u])continue;
        dfs2(v,v);
    }
}
struct Node2{
    int lc,rc,num,lazy;
}t[MAXN*3];
inline void pushdown(int idx,int l,int r)
{
    if(l!=r&&t[idx].lazy)
    {
        t[idx<<1].lazy=t[idx<<1|1].lazy=t[idx<<1].num=t[idx<<1|1].num=1;
        t[idx].lazy=0;
        t[idx<<1].lc=t[idx<<1|1].lc=t[idx<<1].rc=t[idx<<1|1].rc=t[idx].lc;
    }
}
inline void update(int idx)
{
    t[idx].lc=t[idx<<1].lc;
    t[idx].rc=t[idx<<1|1].rc;
    t[idx].num=t[idx<<1].num+t[idx<<1|1].num-
(t[idx<<1].rc==t[idx<<1|1].lc);
}
inline void build(int idx,int l,int r)
{
    t[idx].lazy=0;
    if(l==r)
    {
        t[idx].lc=t[idx].rc=col[inv[l]];
        t[idx].num=1;
        return;
    }
    int mid=(l+r)>>1;
    build(idx<<1,l,mid),build(idx<<1|1,mid+1,r);
    update(idx);
}
```

```

}
inline void color(int idx,int l,int r,int a,int b,int c)
{
    if(l==a&&r==b)
    {
        t[idx].num=t[idx].lazy=1;
        t[idx].lc=t[idx].rc=c;
        return;
    }
    pushdown(idx,l,r);
    int mid=(l+r)>>1;
    if(a>mid)color(idx<<1|1,mid+1,r,a,b,c);
    else if(b<=mid)color(idx<<1,l,mid,a,b,c);
    else
    {
        color(idx<<1|1,mid+1,r,mid+1,b,c);
        color(idx<<1,l,mid,a,mid,c);
    }
    update(idx);
}
inline int query(int idx,int l,int r,int a,int b)
{
    if((l==a&&r==b)||t[idx].lazy)return t[idx].num;
    pushdown(idx,l,r);
    int mid=(l+r)>>1,ans=0;
    if(a>mid)return query(idx<<1|1,mid+1,r,a,b);
    else if(b<=mid)return query(idx<<1,l,mid,a,b);
    else
    {
        ans+=query(idx<<1,l,mid,a,mid);
        ans+=query(idx<<1|1,mid+1,r,mid+1,b);
        ans-=t[idx<<1].rc==t[idx<<1|1].lc;
    }
    return ans;
}
inline int getcolor(int idx,int l,int r,int a)
{
    if(l==r||t[idx].lazy)return t[idx].lc;
    int mid=(l+r)>>1;
    if(a<=mid)return getcolor(idx<<1,l,mid,a);
    else return getcolor(idx<<1|1,mid+1,r,a);
}
inline void solve1(int x,int y,int z)
{
    while(top[x]!=top[y])
    {
        if(dep[top[x]]<dep[top[y]])swap(x,y);
        color(1,1,sz,pos[top[x]],pos[x],z);
        x=father[top[x]];
    }
    if(pos[x]<pos[y])swap(x,y);
}

```

```
    color(1,1,sz,pos[y],pos[x],z);
}
inline void solve2(int x,int y)
{
    ans=0;
    while(top[x]!=top[y])
    {
        if(dep[top[x]]<dep[top[y]])swap(x,y);
        ans+=query(1,1,sz,pos[top[x]],pos[x]);
    }
    if(getcolor(1,1,sz,pos[top[x]])==getcolor(1,1,sz,pos[father[top[x]]]))ans--;
    x=father[top[x]];
}
if(pos[x]<pos[y])swap(x,y);
ans+=query(1,1,sz,pos[y],pos[x]);
printf("%d\n",ans);
}
int main()
{
    memset(head,-1,sizeof(head));
    n=read(),m=read();
    for(RG int i=1;i<=n;++i)col[i]=read();
    int x,y,z;
    for(RG int i=1;i<n;++i)
    {
        x=read(),y=read();
        addedge(x,y),addedge(y,x);
    }
    dfs1(1),dfs2(1,1);
    build(1,1,sz);
    char opt;
    for(RG int i=1;i<=m;++i)
    {
        opt=getchar();
        while(opt!='C'&&opt!='Q')opt=getchar();
        if(opt=='C')
            x=read(),y=read(),z=read(),solve1(x,y,z);
        else if(opt=='Q')
            x=read(),y=read(),solve2(x,y);
    }
    return 0;
}
```

[BZOJ 3531][Sdoi2014]旅行 | 动态开点线段树

code:

```

#include<iostream>
#include<cstdlib>
#include<cstdio>
#include<cstring>
#define MAXN 100005
using namespace std;
int n,q,w[MAXN],c[MAXN],head[MAXN],cnt=0;
int father[MAXN],top[MAXN],maxv[MAXN],siz[MAXN],deep[MAXN],pos[MAXN],sz=0;
int tot=0,rt[MAXN],ls[MAXN*100],rs[MAXN*100],maxn[MAXN*100],sum[MAXN*100];
int read()
{
    int x=0,f=1;char c=getchar();
    while(c<'0' || c>'9'){if(c=='-')f=-1;c=getchar();}
    while(c>='0' && c<='9'){x=x*10+c-'0';c=getchar();}
    return x*f;
}
struct Node1
{
    int next,to;
}Edges[MAXN*2];
void addedge(int u,int v)
{
    Edges[++cnt].next=head[u];
    head[u]=cnt;
    Edges[cnt].to=v;
}
void dfs1(int u)
{
    siz[u]=1;
    for(int i=head[u];~i;i=Edges[i].next)
    {
        int v=Edges[i].to;
        if(v==father[u])continue;
        father[v]=u,deep[v]=deep[u]+1;
        dfs1(v);
        siz[u]+=siz[v];
        if(siz[v]>siz[maxv[u]])maxv[u]=v;
    }
}
void dfs2(int u,int t)
{
    top[u]=t;
    ++sz,pos[u]=sz;
    if(maxv[u])dfs2(maxv[u],t);
    for(int i=head[u];~i;i=Edges[i].next)
    {
        int v=Edges[i].to;
        if(v==maxv[u] || v==father[u])continue;
        dfs2(v,v);
    }
}

```

```
}
void update(int idx)
{
    sum[idx]=sum[ls[idx]]+sum[rs[idx]];
    maxn[idx]=max(maxn[ls[idx]],maxn[rs[idx]]);
}
void change(int &idx,int l,int r,int p,int w)
{
    if(!idx)++tot,idx=tot;
    if(l==r){maxn[idx]=sum[idx]=w;return;}
    int mid=(l+r)>>1;
    if(p<=mid)change(ls[idx],l,mid,p,w);
    else change(rs[idx],mid+1,r,p,w);
    update(idx);
}
int querysum(int idx,int l,int r,int L,int R)
{
    if(!idx)return 0;
    if(L==l&&R==r)return sum[idx];
    int mid=(l+r)>>1;
    if(R<=mid)return querysum(ls[idx],l,mid,L,R);
    else if(L>mid)return querysum(rs[idx],mid+1,r,L,R);
    else return
    querysum(ls[idx],l,mid,L,mid)+querysum(rs[idx],mid+1,r,mid+1,R);
}
int querymax(int idx,int l,int r,int L,int R)
{
    if(!idx)return 0;
    if(L==l&&R==r)return maxn[idx];
    int mid=(l+r)>>1;
    if(R<=mid)return querymax(ls[idx],l,mid,L,R);
    else if(L>mid)return querymax(rs[idx],mid+1,r,L,R);
    else return
    max(querymax(ls[idx],l,mid,L,mid),querymax(rs[idx],mid+1,r,mid+1,R));
}
int QS(int x,int y)
{
    int t=c[x],res=0;
    while(top[x]!=top[y])
    {
        if(deep[top[x]]<deep[top[y]])swap(x,y);
        res+=querysum(rt[t],1,sz,pos[top[x]],pos[x]);
        x=father[top[x]];
    }
    if(deep[x]<deep[y])swap(x,y);
    res+=querysum(rt[t],1,sz,pos[y],pos[x]);
    return res;
}
int QM(int x,int y)
```

```

{
    int t=c[x],res=0;
    while(top[x]!=top[y])
    {
        if(deep[top[x]]<deep[top[y]])swap(x,y);
        res=max(res,querymax(rt[t],1,sz,pos[top[x]],pos[x]));
        x=father[top[x]];
    }
    if(deep[x]<deep[y])swap(x,y);
    res=max(res,querymax(rt[t],1,sz,pos[y],pos[x]));
    return res;
}
int main()
{
    memset(head,-1,sizeof(head));
    n=read(),q=read();
    for(int i=1;i<=n;i++)w[i]=read(),c[i]=read();
    for(int i=1;i<n;i++)
    {
        int x=read(),y=read();
        addedge(x,y),addegedge(y,x);
    }
    deep[1]=1;dfs1(1),dfs2(1,1);
    for(int i=1;i<=n;i++)change(rt[c[i]],1,sz,pos[i],w[i]);
    for(int i=1;i<=q;i++)
    {
        char opt[5];
        scanf("%s",opt);
        int x=read(),y=read();
        if(opt[1]=='C')
        {
            change(rt[c[x]],1,sz,pos[x],0);
            change(rt[y],1,sz,pos[x],w[x]);
            c[x]=y;
        }
        else if(opt[1]=='W')
        {
            change(rt[c[x]],1,sz,pos[x],y);
            w[x]=y;
        }
        else if(opt[1]=='S')
            printf("%d\n",QS(x,y));
        else printf("%d\n",QM(x,y));
    }
    return 0;
}

```

trick | keys:LCA

感觉这两道题有一点点共性但又不是显式的，还是放在一起了，无论如何挺值得两连做的。建议思考，仿佛有智力开发的效果

[BZOJ 3626][LNOI2014]LCA | 线段树

给出 n 个点的有根树 q 次询问，每次询问给出 l, r, z 求 $\sum_{l \leq i \leq r} \text{dep}[\text{LCA}(i, z)]$ $1 \leq n, q \leq 50000$

题解：定义询问 (x, z) 为 $\sum_{l \leq i \leq x} \text{dep}[\text{LCA}(i, z)]$ 则可以将题中询问拆为询问 (r, z) 和 $(l-1, z)$ 的差。于是可以离线，从 1 到 n 逐个加入，加入时将该节点到根节点路径上点的值各加一，线段树维护。而对于每个 z 可以在加入 x 刚好完成时查询 z 到根节点路径上点的值来得到答案。

code:

```
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<cstring>
#include<algorithm>
#define MAXN 50005
#define Mod 201314
using namespace std;
int n, q, deep[MAXN], head[MAXN], cnt=0, tot=0;
int sz=0, maxv[MAXN], father[MAXN], siz[MAXN], top[MAXN], pos[MAXN], res[MAXN*2];
struct Node1
{
    int next, to;
}Edges[MAXN*2];
struct Node2
{
    int l, r, sum, lazy;
}t[MAXN*4];
struct Node3
{
    int x, z, id;
    Node3(int x=0, int z=0, int id=0):x(x), z(z), id(id){}
}Q[MAXN*2];
bool cmp(Node3 a, Node3 b){return a.x<b.x;}
int read()
{
    int x=0, f=1; char c=getchar();
    while(c<'0' || c>'9'){if(c=='-')f=-1; c=getchar();}
    while(c>='0' && c<='9'){x=x*10+c-'0'; c=getchar();}
    return x*f;
}
```

```

void addedge(int u,int v)
{
    Edges[++cnt].next=head[u];
    head[u]=cnt;
    Edges[cnt].to=v;
}
void dfs1(int u)
{
    siz[u]=1;int k=-1;
    for(int i=head[u];~i;i=Edges[i].next)
    {
        int v=Edges[i].to;
        deep[v]=deep[u]+1,dfs1(v);
        siz[u]+=siz[v];
        if(k==-1||siz[v]>siz[k])k=v;
    }
    maxv[u]=k;
}
void dfs2(int u,int t)
{
    ++sz,pos[u]=sz;
    top[u]=t;
    if(maxv[u]!=-1)dfs2(maxv[u],t);
    for(int i=head[u];~i;i=Edges[i].next)
    if(maxv[u]!=Edges[i].to)dfs2(Edges[i].to,Edges[i].to);
}
void build(int idx,int l,int r)
{
    t[idx].l=l,t[idx].r=r;
    if(l==r)return;
    int mid=(l+r)>>1;
    build(idx<<1,l,mid),build(idx<<1|1,mid+1,r);
}
void pushdown(int idx)
{
    if(t[idx].l<t[idx].r&& t[idx].lazy)
    {
        t[idx<<1].lazy+=t[idx].lazy;
        t[idx<<1|1].lazy+=t[idx].lazy;
        t[idx<<1].sum+=t[idx].lazy*(t[idx<<1].r-t[idx<<1].l+1);
        t[idx<<1].sum%=Mod;
        t[idx<<1|1].sum+=t[idx].lazy*(t[idx<<1|1].r-t[idx<<1|1].l+1);
        t[idx<<1|1].sum%=Mod;
        t[idx].lazy=0;
    }
}
void add(int idx,int l,int r)
{
    if(l<=t[idx].l&&r>=t[idx].r)
    {
        t[idx].sum+=t[idx].r-t[idx].l+1;
    }
}

```

```
    t[idx].lazy++;return;
}
pushdown(idx);
int mid=(t[idx].l+t[idx].r)>>1;
if(r<=mid)add(idx<<1,l,r);
else if(l>mid)add(idx<<1|1,l,r);
else add(idx<<1,l,r),add(idx<<1|1,l,r);
t[idx].sum=(t[idx<<1].sum+t[idx<<1|1].sum)%Mod;
}
int query(int idx,int l,int r)
{
    if(l<=t[idx].l&& r>=t[idx].r)return t[idx].sum;
    pushdown(idx);
    int mid=(t[idx].l+t[idx].r)>>1;
    if(r<=mid)return query(idx<<1,l,r);
    else if(l>mid)return query(idx<<1|1,l,r);
    else return (query(idx<<1,l,r)+query(idx<<1|1,l,r))%Mod;
}
void change(int x)
{
    while(top[x])
    {
        add(1,pos[top[x]],pos[x]);
        x=father[top[x]];
    }
    add(1,pos[top[x]],pos[x]);
}
int ask(int x)
{
    int res=0;
    while(top[x])
    {
        res=(res+query(1,pos[top[x]],pos[x]))%Mod;
        x=father[top[x]];
    }
    res=(res+query(1,pos[top[x]],pos[x]))%Mod;
    return res;
}
int main()
{
    memset(head,-1,sizeof(head));
    n=read(),q=read();
    for(int i=1;i<n;i++){father[i]=read();addege(father[i],i);}
    deep[0]=1,dfs1(0);
    dfs2(0,0);build(1,1,sz);
    for(int i=1;i<=q;i++)
    {
        int l=read(),r=read(),z=read();
        ++tot,Q[tot]=Node3(l-1,z,tot);
        ++tot,Q[tot]=Node3(r,z,tot);
    }
}
```

```

}
sort(Q+1,Q+1+tot,cmp);
int now=0;
for(int i=1;i<=tot;i++)
{
    while(now<=Q[i].x)change(now),++now;
    res[Q[i].id]=ask(Q[i].z);
}
for(int i=1;i<=tot;i+=2)
printf("%d\n",(res[i+1]-res[i]+Mod)%Mod);
return 0;
}

```

[BZOJ 4012][HNOI2015]开店 | 主席树

给出 n 个点的带权树，节点 i 的妖怪有年龄 x_i Q 个询问，给出 u, l, r 询问年龄在 $[l, r]$ 的妖怪到 u 点的距离之和，强制在线 $n \leq 150000, Q \leq 200000$ 年龄上限 $A \leq 10^9$

题解：首先点 u, v 间的距离可以转换为 $\text{dis}(u, \text{root}) + \text{dis}(v, \text{root}) - \text{dis}(\text{lca}(u, v), \text{root})$ 对于每个询

问 $\text{dis}(u, \text{root})$ 和 $\sum_{x_v \in [l, r]} \text{dis}(v, \text{root})$ 都是可以通过提前处理快速得到的，那么剩下需要的就是 $\sum_{x_v \in [l, r]} \text{dis}(\text{lca}(u, v), \text{root})$ 这里就是和上一题有点像的地方了，我们可以将年龄离散化然后维护一个主席树从小到大按顺序加入节点，加入时将该节点到根的边权加到主席树里，之后对于每个询问就也可以直接在 $l-1$ 和 r 查询 u 到根得到结果。但主席树的具体过程是有一点玄妙的，还有 lazy tag 什么的想象不出可以见代码 $w[i]$ 维护主席树中节点的父边边权的前缀和。

code:

```

#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<cstring>
#include<algorithm>
#define N 150005
#define Q 200005
using namespace std;
typedef long long LL;
int n, q, A, x[N], y[N], head[N], cnt=0;
int father[N], deep[N], pos[N], top[N], siz[N], maxv[N], sz=0;
int rt[N], ls[N*120], rs[N*120], tot=0;
LL sum[N*120], lazy[N*120], dis[N], d[N], w[N], p1[N], p2[N];
LL read()
{
    LL x=0, f=1; char c=getchar();
    while(c<'0' || c>'9'){if(c=='-')f=-1; c=getchar();}
    while(c>='0' && c<='9'){x=x*10+c-'0'; c=getchar();}
}

```

```
    return x*f;
}
struct Node1
{
    int next,to,w;
}Edges[N*2];
struct Node2
{
    int age,id;
    Node2(int age=0,int id=0):age(age),id(id){}
    bool operator < (const Node2& t) const
    {return age<t.age;}
}data[N];
void addedge(int u,int v,int w)
{
    Edges[++cnt].next=head[u];
    head[u]=cnt;
    Edges[cnt].to=v;
    Edges[cnt].w=w;
}
void dfs1(int u)
{
    siz[u]=1;
    for(int i=head[u];~i;i=Edges[i].next)
    {
        int v=Edges[i].to;
        if(v==father[u])continue;
        father[v]=u;
        deep[v]=deep[u]+1,d[v]=Edges[i].w,dis[v]=dis[u]+Edges[i].w;
        dfs1(v);siz[u]+=siz[v];
        if(!maxv[u]||siz[v]>siz[maxv[u]])maxv[u]=v;
    }
}
void dfs2(int u,int t)
{
    top[u]=t;
    ++sz,pos[u]=sz;
    w[sz]=w[sz-1]+d[u],p1[x[u]]++,p2[x[u]]+=dis[u];
    if(maxv[u])dfs2(maxv[u],t);
    for(int i=head[u];~i;i=Edges[i].next)
    {
        int v=Edges[i].to;
        if(v==maxv[u]||v==father[u])continue;
        dfs2(v,v);
    }
}
void insert(int &idx,int last,int l,int r,int L,int R)
{
    ++tot,idx=tot;
    sum[idx]=sum[last],lazy[idx]=lazy[last],ls[idx]=ls[last],rs[idx]=rs[last];
```

```

    if(L==l&&R==r){lazy[idx]++;sum[idx]+=w[r]-w[l-1];return;}
    int mid=(l+r)>>1;
    if(R<=mid)insert(ls[idx],ls[last],l,mid,L,R);
    else if(L>mid)insert(rs[idx],rs[last],mid+1,r,L,R);
    else
insert(ls[idx],ls[last],l,mid,L,mid),insert(rs[idx],rs[last],mid+1,r,mid+1,
R);
    sum[idx]=sum[ls[idx]]+sum[rs[idx]]+lazy[idx]*(w[r]-w[l-1]);
}
LL query(int s,int t,int l,int r,int L,int R,int tag)
{
    if(L==l&&R==r)return sum[s]-sum[t]+1LL*tag*(w[r]-w[l-1]);
    int mid=(l+r)>>1;tag+=lazy[s]-lazy[t];
    if(R<=mid)return query(ls[s],ls[t],l,mid,L,R,tag);
    else if(L>mid)return query(rs[s],rs[t],mid+1,r,L,R,tag);
    else return
query(ls[s],ls[t],l,mid,L,mid,tag)+query(rs[s],rs[t],mid+1,r,mid+1,R,tag);
}
void add(int i,int x)
{
    while(x)
    {
        insert(rt[data[i].age],rt[data[i].age],1,sz,pos[top[x]],pos[x]);
        x=father[top[x]];
    }
}
LL ask(int l,int r,int u)
{
    LL res=0;
    while(u)
    {
        res+=query(rt[r],rt[l-1],1,sz,pos[top[u]],pos[u],0);
        u=father[top[u]];
    }
    return res;
}
int main()
{
    memset(head,-1,sizeof(head));
    n=read(),q=read(),A=read();
    for(int i=1;i<=n;i++)y[i]=x[i]=read();
    sort(y+1,y+1+n);
    int t=unique(y+1,y+1+n)-y-1;
    for(int i=1;i<=n;i++)x[i]=lower_bound(y+1,y+1+t,x[i])-y;
    for(int i=1;i<n;i++)
    {
        int a=read(),b=read(),c=read();
        addedge(a,b,c),addedge(b,a,c);
    }
    dfs1(1);dfs2(1,1);
    for(int i=1;i<=n;i++)data[i]=Node2(x[i],i);
}

```

```
sort(data+1,data+1+n);
for(int i=1;i<=n;i++)
{
    if(!rt[data[i].age])
        rt[data[i].age]=rt[data[i].age-1];
    add(i,data[i].id);
}
for(int i=1;i<=t;i++)p1[i]+=p1[i-1],p2[i]+=p2[i-1];
LL last=0;
for(int i=1;i<=q;i++)
{
    LL u=read(),a=read(),b=read();
    int l=min((a+last)%A,(b+last)%A),r=max((a+last)%A,(b+last)%A);
    l=lower_bound(y+1,y+1+t,l)-y,r=upper_bound(y+1,y+1+t,r)-y-1;
    if(l>r||l>t||r<1){last=0;printf("0\n");continue;}
    last=dis[u]*(p1[r]-p1[l-1])+p2[r]-p2[l-1]-2*ask(l,r,u);
    printf("%lld\n",last);
}
return 0;
}
```

From: <https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link: https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:wangzai_milk:%E6%A0%91%E9%93%BE%E5%89%96%E5%88%86&rev=1590340287

Last update: 2020/05/25 01:11