主席树

引入

给定一个数列，每次询问[L,R],求区间的kth

如何解决？

暴力：对于每一个询问，排个序，就行了，时间复杂度$O(nmlogn)$

莫队+树状数组：树状数组可以求给定区间kth,使用二分+树状数组，具体不展开，但是多个区间的话，需要不断地进行树状数组的add/del操作，那么使用莫队来优化区间端点的移动问题，时间复杂度$O((n+m)\sqrt{n}logn)$

莫队+平衡树：把树状数组的部分替换成二叉查找树，用splay的一部分操作，需要用到kth操作，不用翻转标记什么的，时间复杂度$O((n+m)\sqrt{n}logn)$跟上面的一样。

以上三种方法都无法解决这道复杂度为$O(nlogn)$的题，因此我们需要引入主席树的概念。

基本概念及实现原理

首先[权值线段树](http://hotpot.clatisus.com/LOTK%ef%bc%88lxh%ef%bc%89/线段树)我们在之前就已经提到，那么主席树就是在每个点建一个权值线段树，而i这个位置的权值线段树里的信息包含[1,i]这个区间内所有的数，这样的话，对于区间[L,R]的询问，我们只需要取L-1,R两颗权值线段树作差处理就能得到每个点的信息了。这样的话查询的时间复杂度是$O(logn)$的

空间处理

然后我们仔细想想，这样的话空间大概是$4n^2$的，显然是不能满足要求的。如何解决？动态开点就行了，每次我们插入一个值，只会改变树上的一条链，如果我们把有更改的点新开点，别的点都直接沿用之前的树已有的点的话，每次只会多$logn$个点，就能够完美的解决复杂度的问题了。

~~~cpp void ins(int &nw,int lst,int l,int r,int x){

```
if(!nw)nw=++cnt;
sum[nw]=sum[lst]+1;
if(l==r)return;
int mid=(l+r)>>1;
if(x<=mid)rs[nw]=rs[lst],ins(ls[nw],ls[lst],l,mid,x);
else ls[nw]=ls[lst],ins(rs[nw],rs[lst],mid+1,r,x);
```

} ~~~

原理简单易懂，下面我们就给出一道板子题作为练手[洛谷P3567 [POI2014]KUR-Couriers](https://www.luogu.com.cn/problem/P3567)

第一眼看过去~~我甚至怀疑我是不是被骗了~~，这不像是主席树的模板题，这不是要维护区间的出现次
~~~

数最多的值吗？一般的主席树能解决？但是我们仔细观察题面，求大于区间数量一半的值，这就意味着，对于一棵主席树上的任意两个点作差关系组成的树，如果存在这样的点，它一定是在每个点左右儿子中总个数大的那个儿子里的，利用这个性质，我们只需要维护每个区间的点个数，找到个数最大点并验证就可以了。

~~~cpp #include<algorithm> #include<stack> #include<ctime> #include<cstring> #include<cmath> #include<iostream> #include<iomanip> #include<cstdio> #include<queue> using namespace std; const int maxn=500005; const int maxt=10000005; 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;
```

} inline void write(int x){

```
if(x/10)write(x/10);
putchar(x%10+'0');
```

} int n,m,tot; int b[maxn],w[maxn]; void bina(){

```
for(int i=1;i<=n;++i)b[++tot]=w[i]=read();
sort(b+1,b+tot+1);
tot=unique(b+1,b+tot+1)-b-1;
for(int i=1;i<=n;++i)w[i]=lower_bound(b+1,b+tot+1,w[i])-b;
```

} int ls[maxt],rs[maxt],sum[maxt]; int root[maxn],cnt; void ins(int &nw,int lst,int l,int r,int x){

```
if(!nw)nw=++cnt;
sum[nw]=sum[lst]+1;
if(l==r)return;
int mid=(l+r)>>1;
if(x<=mid)rs[nw]=rs[lst],ins(ls[nw],ls[lst],l,mid,x);
else ls[nw]=ls[lst],ins(rs[nw],rs[lst],mid+1,r,x);
```

} void Print(int x,int l,int r){

```
printf("%d %d %d\n",sum[x],l,r);
if(l==r)return;
int mid=(l+r)>>1;
Print(ls[x],l,mid);
Print(rs[x],mid+1,r);
```

} void build(){

```
for(int i=1;i<=n;++i)
    ins(root[i],root[i-1],1,tot,w[i]);
```

} int qmax(int rf,int lf,int l,int r,int &x){
~~~

```
if(l==r){x=l;return sum[rf]-sum[lf];}
int num1=sum[ls[rf]]-sum[ls[lf]],num2=sum[rs[rf]]-sum[rs[lf]];
if(num1==num2)return -1;
int mid=(l+r)>>1;
if(num1>num2)return qmax(ls[rf],ls[lf],l,mid,x);
else return qmax(rs[rf],rs[lf],mid+1,r,x);
```

} void query(int l,int r){

```
int tmp;
int num=qmax(root[r],root[l],1,tot,tmp);
if(num*2<=r-l)puts("0");
else write(b[tmp]),puts("");
```

} int main(){

```
n=read();m=read();
bina();build();
for(int i=1,x,y;i<=m;++i){
    x=read();y=read();
    query(x-1,y);
}
return 0;
```

}

~~~

#带修主席树

现在我们给出一道例题[洛谷P2617 Dynamic Rankings](https://www.luogu.com.cn/problem/P2617)

##时间复杂度

一眼看过去，这不就是主席树的题吗？带修？如何解决。

暴力修改？每次要改后面的所有点，修改一次极限时间复杂度$O(nlogn)$~~T的快乐兄弟~~

有什么办法能让我们做更少的修改呢？

我们与树套树的概念相结合，将树状数组和主席树结合起来，树状数组上每个节点挂一颗主席树，这样，我们就得到了插入、查询都是log^2n总体是$O(nlog^2n)$的带修主席树。

##insert

插入上有一点小小的改动，也就是用树状数组的模式插入。

~~~cpp void ins(int &nw,int l,int r,int x,int d){

```
if(!nw)nw=++cnt;
sum[nw]+=d;
if(l==r)return;
int mid=(l+r)>>1;
```

```
if(x<=mid)ins(ls[nw],l,mid,x,d);
else ins(rs[nw],mid+1,r,x,d);
```

} void ins_tr(int x,int v,int d){

```
for(int i=x;i<=n;i+=lowbit(i))ins(tr[i],1,tot,v,d);
```

} ~~~

##query

查询的时候，我们先把所有要用的树状数组的值压入两个数组中，每次查询的时候拿出来用并调整左右儿子就可以了~~ln,rn没清0有40分，谢谢，有被震撼到~~（这个地方数组很多很容易写错，写的时候一定要注意）

~~~cpp void query(int l,int r,int k){

```
if(l==r){write(b[l]);puts("");return;}
int num=0,mid=(l+r)>>1;
for(int i=1;i<=rn;++i)num+=sum[ls[br[i]]];
for(int i=1;i<=ln;++i)num-=sum[ls[bl[i]]];
if(num>=k){
    for(int i=1;i<=rn;++i)br[i]=ls[br[i]];
    for(int i=1;i<=ln;++i)bl[i]=ls[bl[i]];
    query(l,mid,k);
}
else{
    for(int i=1;i<=rn;++i)br[i]=rs[br[i]];
    for(int i=1;i<=ln;++i)bl[i]=rs[bl[i]];
    query(mid+1,r,k-num);
}
```

} void qans(int l,int r,int k){

```
ln=rn=0;
for(int i=l;i>0;i-=lowbit(i))bl[++ln]=tr[i];
for(int i=r;i>0;i-=lowbit(i))br[++rn]=tr[i];
query(1,tot,k);
```

} ~~~

别的部分和一般的主席树没啥区别，下面给出完整代码：

~~~cpp #include<algorithm> #include<stack> #include<ctime> #include<cstring> #include<cmath> #include<iostream> #include<iomanip> #include<cstdio> #include<queue> using namespace std; const int maxn=100005; const int maxt=60000005; 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;
```

} inline void write(int x){

```
if(x/10)write(x/10);
putchar(x%10+'0');
```

} int n,m,tot; int tr[maxn],b[maxn*2],w[maxn]; struct query{

```
bool ty;
int l,r,k,x,y;
```

}q[maxn]; void bina(){

```
sort(b+1,b+tot+1);
tot=unique(b+1,b+tot+1)-b-1;
for(int i=1;i<=n;++i)w[i]=lower_bound(b+1,b+tot+1,w[i])-b;
for(int i=1;i<=m;++i)
    if(q[i].ty)q[i].y=lower_bound(b+1,b+tot+1,q[i].y)-b;
```

} int ls[maxt],rs[maxt],sum[maxt],nw; int lowbit(int x){return x&(-x);} int cnt; void ins(int &nw,int l,int r,int x,int d){

```
if(!nw)nw=++cnt;
sum[nw]+=d;
if(l==r)return;
int mid=(l+r)>>1;
if(x<=mid)ins(ls[nw],l,mid,x,d);
else ins(rs[nw],mid+1,r,x,d);
```

} void ins_tr(int x,int v,int d){

```
for(int i=x;i<=n;i+=lowbit(i))ins(tr[i],1,tot,v,d);
```

} void build(){for(int i=1;i⇐n;++i)ins_tr(i,w[i],1);} int bl[maxn],br[maxn],ln,rn; void query(int l,int r,int k){

```
if(l==r){write(b[l]);puts("");return;}
int num=0,mid=(l+r)>>1;
for(int i=1;i<=rn;++i)num+=sum[ls[br[i]]];
for(int i=1;i<=ln;++i)num-=sum[ls[bl[i]]];
if(num>=k){
    for(int i=1;i<=rn;++i)br[i]=ls[br[i]];
    for(int i=1;i<=ln;++i)bl[i]=ls[bl[i]];
    query(l,mid,k);
}
else{
    for(int i=1;i<=rn;++i)br[i]=rs[br[i]];
    for(int i=1;i<=ln;++i)bl[i]=rs[bl[i]];
    query(mid+1,r,k-num);
```

```
}
```

} void qans(int l,int r,int k){

```
ln=rn=0;
for(int i=l;i>0;i-=lowbit(i))bl[++ln]=tr[i];
for(int i=r;i>0;i-=lowbit(i))br[++rn]=tr[i];
query(1,tot,k);
```

} void solve(){

```
for(int i=1;i<=m;++i)
    if(q[i].ty){
        ins_tr(q[i].x,w[q[i].x],-1);
        ins_tr(q[i].x,w[q[i].x]=q[i].y,1);
    }else qans(q[i].l-1,q[i].r,q[i].k);
```

} int main(){

```
n=read();m=read();
for(int i=1;i<=n;++i)b[++tot]=w[i]=read();
char s[5];
for(int i=1;i<=m;++i){
    scanf("%s",s);
    if(s[0]=='Q'){q[i].ty=0;q[i].l=read();q[i].r=read();q[i].k=read();}
    else {q[i].ty=1;q[i].x=read();b[++tot]=q[i].y=read();}
}
bina();build();solve();
return 0;
```

}

~~~

#树上主席树

树上主席树提出来的一刻相信聪明的各位已经想到怎么做了，我们只要每个点建一颗权值线段树维护它到根的所有值的信息，然后依据题目进行相应的维护就可以了。

下面直接给出一道例题[洛谷P2633 Count on a tree](https://www.luogu.com.cn/problem/P2633)

##insert

我们在建树来维护lca的时候就先把主席树建好

~~~cpp void ins(int &nw,int lst,int l,int r,int x){

```
if(!nw)nw=++cnt;
sum[nw]=sum[lst]+1;
if(l==r)return;
```

```
int mid=(l+r)>>1;
if(x<=mid)rs[nw]=rs[lst],ins(ls[nw],ls[lst],l,mid,x);
else ls[nw]=ls[lst],ins(rs[nw],rs[lst],mid+1,r,x);
```

} void dfs(int x){

```
dep[x]=dep[f[x][0]]+1;
ins(root[x],root[f[x][0]],1,tot,w[x]);
for(unt i=0;i<e[x].size();++i){
    int y=e[x][i];
    if(y==f[x][0])continue;
    f[y][0]=x;
    dfs(y);
}
```

} ~~~

##query

查询的时候我们运用差分，用到lca,f[lca][0]来对答案进行统计

~~~cpp int query(int u,int v,int lca,int plca,int l,int r,int k){

```
if(l==r)return b[l];
int mid=(l+r)>>1;
int num=sum[ls[u]]+sum[ls[v]]-sum[ls[lca]]-sum[ls[plca]];
if(num>=k)return query(ls[u],ls[v],ls[lca],ls[plca],l,mid,k);
else return query(rs[u],rs[v],rs[lca],rs[plca],mid+1,r,k-num);
```

} ~~~

别的就和一般的爬树法和主席树没什么区别了，下面给出完整代码：

~~~cpp #include<algorithm> #include<stack> #include<ctime> #include<cstring> #include<cmath> #include<iostream> #include<iomanip> #include<cstdio> #include<queue> #include<vector> using namespace std; 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;
```

} inline void write(int x){

```
if(x/10)write(x/10);
putchar(x%10+'0');
```

} const int maxn=100005; const int maxt=2000005; int n,m,tot; int b[maxn],w[maxn]; void bina(){

```
sort(b+1,b+tot+1);
tot=unique(b+1,b+tot+1)-b-1;
```

```
for(int i=1;i<=n;++i)w[i]=lower_bound(b+1,b+tot+1,w[i])-b;
```

} #define unt unsigned int vector<int> e[maxn]; int f[maxn][20],dep[maxn],root[maxn]; int ls[maxt],rs[maxt],sum[maxt],cnt; void ins(int &nw,int lst,int l,int r,int x){

```
if(!nw)nw=++cnt;
sum[nw]=sum[lst]+1;
if(l==r)return;
int mid=(l+r)>>1;
if(x<=mid)rs[nw]=rs[lst],ins(ls[nw],ls[lst],l,mid,x);
else ls[nw]=ls[lst],ins(rs[nw],rs[lst],mid+1,r,x);
```

} void dfs(int x){

```
dep[x]=dep[f[x][0]]+1;
ins(root[x],root[f[x][0]],1,tot,w[x]);
for(unt i=0;i<e[x].size();++i){
    int y=e[x][i];
    if(y==f[x][0])continue;
    f[y][0]=x;
    dfs(y);
}
```

} int LCA(int x,int y){

```
if(dep[x]<dep[y])swap(x,y);
for(int i=16;i>=0;--i)
    if(dep[f[x][i]]>=dep[y])x=f[x][i];
if(x==y)return x;
for(int i=16;i>=0;--i)
    if(f[x][i]!=f[y][i])x=f[x][i],y=f[y][i];
return f[x][0];
```

} int query(int u,int v,int lca,int plca,int l,int r,int k){

```
if(l==r)return b[l];
int mid=(l+r)>>1;
int num=sum[ls[u]]+sum[ls[v]]-sum[ls[lca]]-sum[ls[plca]];
if(num>=k)return query(ls[u],ls[v],ls[lca],ls[plca],l,mid,k);
else return query(rs[u],rs[v],rs[lca],rs[plca],mid+1,r,k-num);
```

} void solve(){

```
int lastans=0;
for(int i=1,u,v,k;i<=m;++i){
    u=read()^lastans;v=read();k=read();
    int lca=LCA(u,v);
    write(lastans=query(root[u],root[v],root[lca],root[f[lca][0]],1,tot,k));
    puts("");
```

```
}
```

} int main(){

```
n=read();m=read();
for(int i=1;i<=n;++i)b[++tot]=w[i]=read();
bina();
for(int i=1,x,y;i<n;++i){
    x=read();y=read();
    e[x].push_back(y);
    e[y].push_back(x);
}
dfs(1);
for(int i=1;i<=16;++i)
    for(int j=1;j<=n;++j)
        f[j][i]=f[f[j][i-1]][i-1];
solve();
return 0;
```

}

~~~

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

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:hotpot:lotk_%E4%B8%BB%E5%B8%AD%E6%A0%91

Last update: **2020/05/08 16:28**
~~~