

Codeforces Round #665 (Div. 2)

[比赛链接](#)

E. Divide Square

题意

给定一个 $10^6 \times 10^6$ 的正方形，接下来给定 n 条平行于 X 轴的线段和 m 条平行于 Y 轴的线段。

问正方形被线段划分为多少区域。数据保证所有线段不共线且至少与正方形边界交于一个点。

题解

有结论正方形被线段划分的区域数等于 $1 +$ 同时与正方形左右边界或上下边界相交的线段数 $+$ 所有线段的交点数。

然后线段树维护所有线段的交点数即可。

```
const int MAXN=1e5+5,M=1e6,M2=1e6+5;
struct Line{
    int y,a,b;
    bool operator < (const Line &b) const{
        return y<b.y;
    }
}OX[MAXN],OY[M2<<1];
int lef[M2<<2],rig[M2<<2],s[M2<<2];
void build(int k,int L,int R){
    lef[k]=L,rig[k]=R;
    if(L==R)return;
    int M=L+R>>1;
    build(k<<1,L,M);build(k<<1|1,M+1,R);
}
void update(int k,int pos,int v){
    s[k]+=v;
    if(lef[k]==rig[k])return;
    int mid=lef[k]+rig[k]>>1;
    if(mid>=pos)update(k<<1,pos,v);
    else update(k<<1|1,pos,v);
}
int query(int k,int L,int R){
    if(L<=lef[k]&&rig[k]<=R)return s[k];
    int ans=0,mid=lef[k]+rig[k]>>1;
    if(mid>=L)ans+=query(k<<1,L,R);
    if(mid<R)ans+=query(k<<1|1,L,R);
}
```

```
return ans;
}
int main()
{
    int n=read_int(),m=read_int();
    LL ans=1;
    _for(i,0,n){
        OX[i].y=read_int(),OX[i].a=read_int(),OX[i].b=read_int();
        if(OX[i].b-OX[i].a==M)ans++;
    }
    _for(i,0,m){
        OY[i].a=OY[i+m].a=read_int(),OY[i].y=read_int(),OY[i+m].y=read_int();
        if(OY[i+m].y-OY[i].y==M)ans++;
        OY[i].b=1,OY[i+m].b=-1,OY[i+m].y++;
    }
    m<=<1;
    sort(OX,OX+n);sort(OY,OY+m);
    build(1,0,M);
    int pos=0;
    _for(i,0,n){
        while(pos<m&&OY[pos].y<=OX[i].y)
            update(1,OY[pos].a,OY[pos].b),pos++;
        ans+=query(1,OX[i].a,OX[i].b);
    }
    enter(ans);
    return 0;
}
```

F. Reverse and Swap

题意

给定一个长度为 2^n 的序列 $\{a\}$ 接下来 q 次操作：

1. 将 a_x 的值修改为 k
2. 将所有 $[(i-1)2^{k+1}, i2^k]$ 区间翻转
3. 将所有 $[(2i-2)2^{k+1}, (2i-1)2^k]$ 区间与 $[(2i-1)2^{k+1}, (2i)2^k]$ 区间交换
4. 询问 $[l, r]$ 区间的和

题解

将 $\{a\}$ 下标改为从零开始。不难发现操作 2 相当于将 a_x 变为 $a_{x \oplus (2^k-1)}$ 操作 3 相当于将 a_x 变为 $a_{x \oplus 2^k}$

于是对于操作 2 和操作 3，只需要维护最终下标的异或值 val 即可，对于操作 1，只需要修改 $a_{x \oplus \text{val}}$ 即可。

对于操作 \$4\$，不妨先将查询区间分割为若干段形如 $[M2^k, M2^k+2^{k-1}]$ 的区间。

不难发现 $[0, 2^{k-1}]$ 区间异或 $\text{val} \wedge (2^{k-1})$ 后仍然遍历 $[0, 2^{k-1}]$

于是 $[M2^k, M2^k+2^{k-1}]$ 异或 val 后变为 $[M2^k \oplus (\text{val} \wedge (2^{k-1})), M2^k \oplus (\text{val} \wedge (2^{k-1})) + 2^{k-1}]$ 考虑线段树查询即可。

总时间复杂度 $O(2^n + nq)$

```
const int MAXN=1<<18;
int a[MAXN],val;
LL s[MAXN<<2];
void build(int k,int L,int R){
    int M=L+R>>1;
    if(L==R)return s[k]=a[M],void();
    build(k<<1,L,M);build(k<<1|1,M+1,R);
    s[k]=s[k<<1]+s[k<<1|1];
}
void update(int k,int L,int R,int pos,int v){
    if(L==R)return s[k]=v,void();
    int M=L+R>>1;
    if(M>=pos)update(k<<1,L,M,pos,v);
    else update(k<<1|1,M+1,R,pos,v);
    s[k]=s[k<<1]+s[k<<1|1];
}
LL query(int k,int L,int R,int ql,int qr,int pos){
    if(ql<=L&&R<=qr)return s[k];
    int M=L+R>>1,dir=(val>>pos)&1;
    if(M>=qr)return query((k<<1)^dir,L,M,ql,qr,pos-1);
    else if(M<ql)return query((k<<1|1)^dir,M+1,R,ql,qr,pos-1);
    else return
    query((k<<1)^dir,L,M,ql,qr,pos-1)+query((k<<1|1)^dir,M+1,R,ql,qr,pos-1);
}
int main()
{
    int n=read_int(),q=read_int(),m=1<<n;
    _for(i,0,m)a[i]=read_int();
    build(1,0,m-1);
    while(q--){
        int opt=read_int(),x=read_int();
        if(opt==1)
            update(1,0,m-1,(x-1)^val,read_int());
        else if(opt==2)
            val^=(1<<x)-1;
        else if(opt==3)
            val^=1<<x;
        else
            enter(query(1,0,m-1,x-1,read_int()-1,n-1));
    }
    return 0;
}
```

Last
update:
2020/08/23 11:58 2020-2021:teams:legal_string:jxm2001:contest:cf_665_div_2 https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:cf_665_div_2

}

From:

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

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:cf_665_div_2 

Last update: **2020/08/23 11:58**