

Educational Codeforces Round 92

[比赛链接](#)

D. Segment Intersections

题意

给定两种线段 $[l_a, r_a], [l_b, r_b]$ 每种线段有 n 条，记为 $a_1, a_2 \dots a_n$ 和 $b_1, b_2 \dots b_n$

每次操作可以任选一条线段 $[l, r]$ 将其变换成 $[l-1, r]$ 或 $[l, r+1]$

要求输出最小操作步数，使得 $\sum_{i=1}^n f(a_i, b_i) \geq k$ 其中 $f(a, b)$ 表示线段 a, b 相交部分长度。

题解 1

暴力枚举 i 只考虑前 i 对线段求出最小答案，时间复杂度 $O(n)$

```
int main()
{
    int t=read_int();
    while(t--){
        LL
n=read_int(),k=read_int(),l1=read_int(),r1=read_int(),l2=read_int(),r2=read_int(),ans=1e10;
        LL tot=max(r1,r2)-min(l1,l2),d=max(l1,l2)-min(r1,r2);
        _rep(i,1,n){
            if(tot*i>=k)
                ans=min(ans,d*i+k);
            else
                ans=min(ans,(tot+d)*i+(k-tot*i)*2);
        }
        ans=max(ans,0LL);
        enter(ans);
    }
    return 0;
}
```

题解 2

分类讨论，时间复杂度 $O(1)$

```
int main()
{
    int t=read_int();
    while(t--){
        LL
n=read_int(),k=read_int(),l1=read_int(),r1=read_int(),l2=read_int(),r2=read_int();
        if(l1>l2)
            swap(l1,l2),swap(r1,r2);
        if(r1>=r2){
            k-=n*(r2-l2);
            r1-=(r2-l2);
            r2=l2;
            swap(r1,r2);
        }
        else if(r1>=l2){
            k-=n*(r1-l2);
            swap(r1,l2);
            r2-=l2-r1;
            l2=r1;
        }
        if(k<=0){
            puts("0");
            continue;
        }
        int cost=2*(l2-r1),d=l2-r1,len1=r1-l1,len2=r2-l2;
        if(len1+len2+d==0){
            enter(k*2);
            continue;
        }
        if(k>=n*(len1+len2+d)){
            k-=n*(len1+len2+d);
            enter(n*(len1+len2+cost)+2*k);
            continue;
        }
        int ks=k/(len1+len2+d);
        k%=(len1+len2+d);
        if(ks>0)
            enter(ks*(len1+len2+cost)+min(2*k,d+k));
        else
            enter(d+k);
    }
    return 0;
}
```

E. Calendar Ambiguity

题意

求满足同余方程 $xd+y\equiv yd+x\pmod w$ 其中 $1\leq x,y\leq n$

题解

移项，得 $d(x-y)\equiv 0\pmod w$ 记 $w'=\frac{w}{(w,d)}$ 于是问题等价于 $w'\mid(x-y)$

考虑枚举 $(x-y)=iw'$ 于是满足条件的解个数为 $\sum_{i=1}^{\lfloor\frac{n}{w'}\rfloor} n-iw'=n\lfloor\frac{n}{w'}\rfloor-\frac{\lfloor\frac{n}{w'}\rfloor(\lfloor\frac{n}{w'}\rfloor+1)}{2}$

时间复杂度 $O(\log(\min(w,d)))$

```
int main()
{
    int t=read_int();
    while(t--){
        int m=read_int(),d=read_int(),w=read_int(),n,k;
        w/=__gcd(w,d-1);
        n=min(m,d),k=n/w;
        enter(1LL*n*k-1LL*(1+k)*k*w/2);
    }
    return 0;
}
```

F. Bicolored Segments

题意

给定 n 条线段 $[l_i,r_i]$ 每种线段有黑白两种颜色。要求选出尽量多的线段，使得异色线段不相交。

题解 1

考虑将异色相交的线段间连一条线，则原题转化为二分图，答案等价于最大独立集。

而二分图的最大独立集等价于总节点点数减去二分图最大匹配，于是问题转化为求二分图最大匹配。

读入所有线段，把每条线段拆成 l_i 加入和 r_i 删除两个事件。

把所有事件按 x 轴位置排序，如果处于 x 轴同一位置，则加入事件优先于删除事件。按顺序扫描 x 轴并处理事件。

扫描过程中维护两种颜色的集合，每种颜色的集合维护扫描到当前位置时已经加入且未被删除的线段，且集合中线段按右端点从小到大排序。

扫描到加入事件，则将该事件对应线段加入对应颜色的集合。如果扫描到删除事件且该事件对应线段已经配对则跳过。

否则考虑从异色集合中找到一条线段和该线段配对，显然配对的线段右端点越小越优。

考虑该贪心策略的正确性，假设现在考虑的线段是 u_1 被配对的线段是 v_1

情况一，假设更优策略中 u_1 未配对，则令 u_1 与 v_1 配对，最多导致原来与 v_1 配对的点无法配对，答案不变，假设不成立。

情况二，假设更优策略中 v_1 未配对，则令 u_1 与 v_1 配对，最多导致原来与 u_1 配对的点无法配对，答案不变，假设不成立。

情况三，假设更优策略中 u_1 与 v_2 配对 u_2 与 v_1 配对。

于是根据假设有 u_1 删除前 v_2 已经加入 v_1 删除前 u_2 已加入。(这里删除指遇到删除事件)

根据当前决策方式有 u_1 删除前 u_2 未删除 v_1 删除前 v_2 未删除。

于是 u_2 与 v_2 可以配对，且如果该两点配对，则答案不变，假设不成立。

于是无论如何该贪心算法总是成立。


```
const int MAXN=2e5+5;
int lef[MAXN],rig[MAXN],c[MAXN];
struct Event{
    int x,v,id;
    bool operator < (const Event &b)const{
        return x<b.x||(x==b.x&&v>b.v);
    }
}a[MAXN<<1];
set<pair<int,int> > s[2];
int main()
{
    int n=read_int();
    _for(i,0,n){
        lef[i]=read_int(),rig[i]=read_int(),c[i]=read_int()-1;
        a[i]=Event{lef[i],1,i};
        a[i+n]=Event{rig[i],-1,i};
    }
    sort(a,a+2*n);
    int ans=n;
    _for(i,0,2*n){
        int pos=a[i].id;
        if(a[i].v>0)
            s[c[pos]].insert(make_pair(rig[pos],pos));
        else{
            if(s[c[pos]].find(make_pair(rig[pos],pos))!=s[c[pos]].end()){
                s[c[pos]].erase(make_pair(rig[pos],pos));
            }
        }
    }
}
```

```
        if(s[c[pos]^1].begin() != s[c[pos]^1].end()){
            s[c[pos]^1].erase(s[c[pos]^1].begin());
            ans--;
        }
    }
}
enter(ans);
return 0;
}
```

From:

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

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:edu_92&rev=1596185326 

Last update: **2020/07/31 16:48**