CCPC Wannafly Camp Day5

比赛链接

B. Bitset Master

题意

给定 n 阶树和 m 个操作。每个点 u 维护一个集合 $S(u)$，初始时 $S(u)=\{u\}$。

每次操作选定一条树边 $u\to v$，令 $S(u)=S(v)=\text{old}(S(u)\cup S(v))$。

对每个 u，输出有多少个 v 满足 $u\in S(v)$。

题解

设 m 次操作选择的边为 $e_1,e_2\cdots e_m$，观察 $u\in S(v)$，这等价于序列 $\{e\}$ 中存在某个子序列 $e_{p_1},e_{p_2}\cdots e_{p_k}$ 恰好是连接 u,v 的路径。

如果逆序操作，即序列变为 $e_{p_k},e_{p_{k-1}}\cdots e_{p_1}$，则有 $v\in S(u)$。

于是正序操作 $\sum_{v=1}^n [u\in S(v)]=$ 逆序操作 $\sum_{v=1}^n [v\in S(u)]=$ 逆序操作 $|S(u)|$。

接下来考虑怎样维护 $|S(u)|$，根据容斥定理，有 $|S(u)\cup S(v)|=|S(u)|+|S(v)|-|S(u)\cap S(V)|$。

对 $t\in S(u)\cap S(v)$，由于树形结构约束，必存在子序列 $e_{p_1},e_{p_2}\cdots e_{p_k}$ 代表路径 $k\to \cdots v\to u$ 或 $k\to \cdots u\to v$。

于是不难发现 $|S(u)\cap S(V)|$ 恰好为上次合并 $S(u),S(v)$ 的结果，维护一下即可。时间复杂度 $O(n+m)$。

```
const int MAXN=5e5+5;
struct Edge{
    int u,v,w;
}edge[MAXN];
int c[MAXN],p[MAXN];
int main()
{
    int n=read_int(),m=read_int();
    _rep(i,1,n)c[i]=1;
    _for(i,1,n){
        int u=read_int(),v=read_int();
        edge[i]=Edge{u,v,0};
    }
    _for(i,0,m)
    p[i]=read_int();
```

```
    for(int i=m-1;i>=0;i--){
        int u=edge[p[i]].u,v=edge[p[i]].v,cc=c[u]+c[v]-edge[p[i]].w;
        c[u]=cc,c[v]=cc,edge[p[i]].w=cc;
    }
    _rep(i,1,n)
    space(c[i]);
    return 0;
}
```

C. Self-Adjusting Segment Tree

题意

给定线段树的 m 次区间查询操作，要求构造一棵维护区间 $[1,n]$ 的特殊线段树。

使得查询操作访问结点的次数最少的结点并输出访问结点的次数。

题解

不难得到 dp 方程

$$ \text{dp}(i,j)=\min(\text{dp}(i,k)+\text{dp}(k+1,j)+w(i,j)) $$

然后考虑依次处理每个询问 $[ql,qr]$ 对 $w(i,j)$ 的贡献。不难发现当 $[i,j]$ 与 $[ql,qr]$ 相交但 $[i,j]\not\subseteq [ql,qr]$ 时询问贡献为 1。

问题在于 $[i,j]\subseteq [ql,qr]$ 询问对 $w(i,j)$ 的贡献与询问是否包含 $[i,j]$ 区间的祖结点有关。

有结论任意二叉树的叶子结点数 $=$ 非叶子结点数 $+1$，而线段树恰好为二叉树。

于是不妨转化思路令 $[i,i]\subseteq [ql,qr]$ 得到贡献 1，令 $[i,j](i\lt j)\subseteq [ql,qr]$ 得到贡献 -1。

于是 dp 过程中如果划分区间得到 $[i,j]\subseteq [ql,qr]$则 $[i,j]$ 的子树的总贡献恰好为 1。

对应每个询问对 $w(i,j)$ 的贡献，差分维护即可，时间复杂度 $O(n^3+m)$

```
const int MAXN=505;
const LL inf=1e16;
LL d1[MAXN],d2[MAXN][MAXN],w[MAXN][MAXN],dp[MAXN][MAXN];
void update(int l1,int l2,int r1,int r2,int v){
    d2[l1][r1]+=v;
    d2[l1][r2+1]-=v;
    d2[l2+1][r1]-=v;
    d2[l2+1][r2+1]+=v;
}
int main()
{
```

```
    int n=read_int(),m=read_int();
    while(m--){
        int ql=read_int(),qr=read_int();
        update(1,ql-1,ql,qr,1);
        update(1,qr,qr+1,n,1);
        update(ql,qr,ql,qr,-1);
        d1[ql]+=2;
        d1[qr+1]-=2;
    }
    _rep(i,1,n)_rep(j,1,n)
    w[i][j]=d2[i][j]+w[i-1][j]+w[i][j-1]-w[i-1][j-1];
    _rep(i,1,n){
        d1[i]+=d1[i-1];
        w[i][i]+=d1[i];
    }
    _rep(i,1,n)_rep(j,1,n){
        if(i==j)
        dp[i][j]=w[i][j];
        else
        dp[i][j]=inf;
    }
    for(int i=n;i;i--)_rep(j,i+1,n)_for(k,i,j)
    dp[i][j]=min(dp[i][j],dp[i][k]+dp[k+1][j]+w[i][j]);
    enter(dp[1][n]);
    return 0;
}
```

J. Xor on Figures

题意

给定 $2^k\times 2^k$ 的一个二维 01 串。定义 $B(x,y)$ 表示将二维 01 串循环左移 x 格循环上移 y 格的新二维 01 串。

定义两个二维串之间的异或为对应位置异或。问任意个 $B(x_i,y_i)$ 异或能得到的所有不同的二维 01 串个数。

题解

直接把 $B(x,y)$ 拉成一维，然后高斯消元计算所有 2^{2k} 个 $B(x,y)$ 的秩即可。时间复杂度 $I\left(2^{4k}+\frac {2^{6k}}w\right)$

```
const int MAXN=35,Mod=1e9+7;
int a[MAXN][MAXN];
bitset<MAXN*MAXN> x[MAXN*MAXN];
char s[MAXN];
int quick_pow(int a,int k){
```

```
    int ans=1;
    while(k){
        if(k&1)ans=1LL*ans*a%Mod;
        a=1LL*a*a%Mod;
        k>>=1;
    }
    return ans;
}
int main()
{
    int k=read_int(),n=1<<k;
    _for(i,0,n){
        scanf("%s",s);
        _for(j,0,n)
        a[i][j]=s[j]-'0';
    }
    _for(di,0,n)_for(dj,0,n)
    _for(i,0,n)_for(j,0,n)
    x[di*n+dj][i*n+j]=a[(i+di)%n][(j+dj)%n];
    n*=n;
    int rk=0;
    _for(i,0,n){
        if(!x[i][i]){
            _for(j,i+1,n){
                if(x[j][i]){
                    swap(x[i],x[j]);
                    break;
                }
            }
        }
        if(x[i][i])
        rk++;
        _for(j,i+1,n){
            if(x[j][i])
            x[j]^=x[i];
        }
    }
    enter(quick_pow(2,rk));
    return 0;
}
```