

CCPC Wannafly Camp Day5

[比赛链接](#)

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 $[i, j] (i < j) \subseteq [ql, qr]$ 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 $\left(2^{4k} + \frac{2^{6k}}{w}\right)$

```
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;
}
```

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

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:ccpc_wannafly_winter_camp_day5&rev=1622036156

Last update: 2021/05/26 21:35