FWT刷题

CF662C

题意：给定一个n*m的01矩阵，可以取反若干次任意行或列，问若干次操作之后1最少是多少。

数据范围$n\leq 20,m\leq 100000$

题解n的范围很小，我们可以考虑二进制枚举行的取反策略，假设行的翻转策略为S那么这道题目的答案就是$\sum_{i=1}^m min(f(S\oplus a_i),n-f(S\oplus a_i))$,f函数为这一列1的个数。

那么我们假设$g(i)=min(f(i),n-f(i))$那么原式就可以改写为$\sum_{j,i\oplus S = j}g(j)\times f(i)$,按照异或的性质，可以直接改写为

$\sum_{j,i\oplus j = S}g(j)\times f(i)$

那么好了，接下来我们就可以轻松的用fwt解决这道题目了。

```
#include <bits/stdc++.h>
using namespace std;
const int V = 1<<20;
const int N = 22;
const int M = 100005;

int n,a[M];
long long dp[V],cnt[V];
char s[N][M];
int calc(int x) {
    int ans = 0;
    while (x) {
        ans+=(x&1);
        x>>=1;
    }
    return ans;
}

void FWT(long long *src,int len) {
    for (int sz = 2;sz <= len; sz<<=1) {
        int step = sz>>1;
        for (int i = 0;i < len;i+=sz) {
            for (int j = i;j < i+step;j++) {
                long long a = src[j],b = src[j+step];
                src[j] = a+b;
                src[j+step] = a-b;
            }
        }
    }
}
```

```
void IFWT(long long *src,int len) {
    for (int sz = 2;sz <= len;sz <<= 1) {
        int step = sz >> 1;
        for (int i = 0;i<len;i+=sz) {
            for (int j = i;j < i+step;j++) {
                long long a = src[j],b = src[j+step];
                src[j] = (a+b)>>1;
                src[j+step] = (a-b)>>1;
            }
        }
    }
}
int main()
{
    int n,m;
    scanf("%d%d",&n,&m);
    for (int i = 0;i<n;i++)
        scanf("%s",s[i]+1);
    for (int i = 1;i<= m;i++) {
        int tmp = 0;
        for (int j = 0;j < n;j++)
            tmp |= ((s[j][i]-'0')<<j);
        cnt[tmp]++;
    }
    int len = 1<<n;
    for (int i = 0;i<len;i++)
    {
        int tmp = calc(i);
        dp[i] = min(tmp,n-tmp);
    }
    FWT(cnt,len);
    FWT(dp,len);
    for (int j = 0;j < len;j++)
        dp[j] *= cnt[j];
    IFWT(dp,len);
    long long ans = n*m+1;
    for (int j = 0;j < len;j++)
        ans = min(ans,dp[j]);
    printf("%lld\n",ans);
    return 0;
}
```

CF914G

HDU5909

给定一个n个点的树，定义一个树的权值是书上所有点点权的异或值，问这个树权值为$[0,m)$的子图分别有多少个。

直接树形dp然后用fwt加速计算就行惹

```
#include <bits/stdc++.h>
using namespace std;
const int V = 1<<20;
const int N = 22;
const int M = 100005;

int n,a[M];
long long dp[V],cnt[V];
char s[N][M];
int calc(int x) {
    int ans = 0;
    while (x) {
        ans+=(x&1);
        x>>=1;
    }
    return ans;
}

void FWT(long long *src,int len) {
    for (int sz = 2;sz <= len; sz<<=1) {
        int step = sz>>1;
        for (int i = 0;i < len;i+=sz) {
            for (int j = i;j < i+step;j++) {
                long long a = src[j],b = src[j+step];
                src[j] = a+b;
                src[j+step] = a-b;
            }
        }
    }
}

void IFWT(long long *src,int len) {
    for (int sz = 2;sz <= len;sz <<= 1) {
        int step = sz >> 1;
        for (int i = 0;i<len;i+=sz) {
            for (int j = i;j < i+step;j++) {
                long long a = src[j],b = src[j+step];
                src[j] = (a+b)>>1;
                src[j+step] = (a-b)>>1;
            }
        }
    }
}
```

```
int main()
{
    int n,m;
    scanf("%d%d",&n,&m);
    for (int i = 0;i<n;i++)
        scanf("%s",s[i]+1);
    for (int i = 1;i<= m;i++) {
        int tmp = 0;
        for (int j = 0;j < n;j++)
            tmp |= ((s[j][i]-'0')<<j);
        cnt[tmp]++;
    }
    int len = 1<<n;
    for (int i = 0;i<len;i++)
    {
        int tmp = calc(i);
        dp[i] = min(tmp,n-tmp);
    }
    FWT(cnt,len);
    FWT(dp,len);
    for (int j = 0;j < len;j++)
        dp[j] *= cnt[j];
    IFWT(dp,len);
    long long ans = n*m+1;
    for (int j = 0;j < len;j++)
        ans = min(ans,dp[j]);
    printf("%lld\n",ans);
    return 0;
}
```

HDU5823

HDU6057