

状压dp练习2

人员

于潇涵、王梓同、蔡云翔、石宇赫、李佳声、王崇宇、窦浩轩、胡赫轩、崔嘉睿、穆鹏宇、程晟泰、梁钰涵 到课

作业检查

于潇涵 上周请假

王梓同 未完成

蔡云翔 已完成

石宇赫 已完成

李佳声 已完成

王崇宇 已完成

窦浩轩 上周请假

胡赫轩 已完成

崔嘉睿 已完成

穆鹏宇 已完成

程晟泰 已完成

梁钰涵 已完成

作业

<https://www.luogu.com.cn/contest/169473>

A、B、C 3道题目

(A、B题目要求必须通过，C题必须要思考并实现，不通过也没关系，下周会讲，但是必须要做)

课堂表现

同学们整体基础不是很好，课下一定要好好练习刷题题单

课堂内容

Minimax Problem

题目要求最小值最大，一眼二分，二分最终最小的答案

因为 m 非常小，**check**时可以把 m 列的状态通过状态压缩的方式映射成一个数

开一个 f 数组，记录一个状态对应的行

之后利用 $256 * 256$ 的时间复杂度统计是否有合法解即可

```

#include <bits/stdc++.h>

using namespace std;

const int maxn = 3e5 + 5, N = 8;
int w[maxn][N];

bool check(int n, int m, int mid, int& ansL, int& ansR) {
    int f[1<<N];
    memset(f, 0, sizeof(f));
    for (int i = 1; i <= n; ++i) {
        int res = 0;
        for (int j = 0; j < m; ++j) {
            if (w[i][j] >= mid) res |= (1<<j);
        }
        f[res] = i;
    }

    for (int i = 0; i < (1<<m); ++i) {
        for (int j = 0; j < (1<<m); ++j) {
            if ((i|j)==(1<<m)-1 && f[i] && f[j]) {
                ansL = f[i], ansR = f[j];
                return true;
            }
        }
    }
    return false;
}

int main()
{
    int n, m; cin >> n >> m;
    for (int i = 1; i <= n; ++i) {
        for (int j = 0; j < m; ++j) cin >> w[i][j];
    }

    int l = 0, r = 1e9;
    int ansL = 1, ansR = 1;
    while (l <= r) {
        int mid = (l + r) / 2;
        if (check(n, m, mid, ansL, ansR)) l = mid+1;
        else r = mid-1;
    }
    cout << ansL << " " << ansR << endl;
    return 0;
}

```

Petya and Spiders

定义状态 $f[i][j][k]$:

第 i 行状态为 j , 第 $i-1$ 行状态为 k 时, 前 $i-1$ 行最多有多少空白格

转移过程: $f[i][j][k] \leftarrow f[i-1][k][l] + \text{sum}(k)$

$\text{sum}(k)$: k 这个状态代表第 $i-1$ 行有几个空白格

转移条件: 对于 k 每个为 0 的二进制位, 要求它的 左边、右边、上边、下边 至少存在一个 1

```

#include <bits/stdc++.h>

using namespace std;

const int N = 40 + 5, M = 6;
int f[N][1<<M][1<<M];
// f[i, j, k]: i ~ j, i-1 ~ k

bool check(int n, int m, int i, int j, int k, int l) {
    // i: j i-1: k i-2: l
    if (i-2==0 && l) return false;
    if (i==n+1 && j) return false;
    for (int p = 0; p < m; ++p) {
        if (!(k>>p) & 1)) {
            if ((j>>p) & 1) continue;
            if ((l>>p) & 1) continue;
            if (p>0 && ((k>>(p-1)) & 1)) continue;
            if (p<m && ((k>>(p+1)) & 1)) continue;
            return false;
        }
    }
    return true;
}

int sum(int k, int m) {
    int res = 0;
    for (int i = 0; i < m; ++i) {
        if (!(k>>i) & 1)) ++res;
    }
    return res;
}

int main()
{
    int n, m; cin >> n >> m;
    if (n < m) swap(n, m);

    int res = 0;
    for (int i = 2; i <= n+1; ++i) {
        for (int j = 0; j < (1<<m); ++j) {
            for (int k = 0; k < (1<<m); ++k) {
                for (int l = 0; l < (1<<m); ++l) {
                    // i: j i-1: k i-2: l
                    if (check(n, m, i, j, k, l)) {
                        f[i][j][k] = max(f[i][j][k], f[i-1][k][l] + sum(k, m));
                        if (i == n+1) res = max(res, f[i][j][k]);
                    }
                }
            }
        }
    }
}

```

```
}  
  
cout << res << endl;  
return 0;  
}
```